Reliability of Marine Structures Program

SECOND-ORDER RANDOM OCEAN WAVES: PREDICTION OF TEMPORAL AND SPATIAL VARIATION FROM FIXED AND MOVING REFERENCES

THE ROUTINE WAVEMAKER

Version 3.2

Bert Sweetman

Steven R. Winterstein

Civil Engineering Department, Stanford University

Supported by

Offshore Technology Research Center

Office of Naval Research

Report No. RMS–37

May 1999

20011123 057

Department of CIVIL ENGINEERING
STANFORD UNIVERSITY

Version History

This release of WAVEMAKER software incorporates recent technical improvements achieved at the Reliability of Marine Structures Program: An optional high frequency cut-off has been implemented on the first-order part of the wave simulation which allows the user to control the highest frequency at which energy is assigned by the specified wave spectrum. Also, the method by which end-point discontinuities are treated in the wave identification option has been significantly enhanced.

WAVEMAKER 3.2 is backward compatible with WAVEMAKER 3.0 and 3.1, except for use of *omgmax*. Results from a run of WAVEMAKER 3.0 or 3.1 can be exactly obtained with this new release as long as *omgmax* is not applied. However, the user is reminded that WAVEMAKER 3.0 is not fully backward compatible with earlier versions because there were some minor improvements to existing parts of the program. Input files to all earlier versions can be directly used with this new version.

A version history of WAVEMAKER follows:

- **Version 1.0**: Released in April 1995 and documented in Jha and Winterstein, 1995, Report RMS–17, contains simulation capabilities for second-order random waves.
- **Version 1.1**: Released in August 1995, includes modification of temporary intermediate output file to use less disk space and to reduce program execution time by approximately 50%. Additionally, a DOS executable of WAVEMAKER was included in this release.
- **Version 2.0**: Released June 1996, includes identification capabilities so that underlying first-order wave history can be retrieved from an observed wave history, Report RMS–22 (Jha and Winterstein, 1996)
- **Version 3.0**: Released June 1998, includes prediction capabilities so that wave histories at other spatial locations can be predicted from a user-specified wave history at a known location. Also, endpoint discontinuities in the wave records are treated prior to application of FFT's, Report RMS–33 (Sweetman, Jha and Winterstein, 1998).

i

- **Version 3.1**: Released September 1998, includes prediction capabilities so that wave histories can be predicted as observed from a moving reference point with a known motion time history based on an observed wave history at a fixed spatial location, Technical Note TN–5 (Sweetman, and Winterstein, 1998).
- **Version 3.2**: Released May 1999. High frequency cut-off modified; now applied to first-order part of simulated waves only. Treatment of endpoint discontinuities in *identify* modified, Report RMS–38.

Acknowledgments

iv

Table of Contents

List of Figures

Abstract

WAVEMAKER is a FORTRAN program used to simulate random non-Gaussian ocean wave histories, to identify the underlying first- and second-order components of user specified waves, or to predict wave time histories at other user specified fixed or moving spatial locations based on the originally simulated or identified wave history.

Ocean wave histories are simulated by generating a first-order (Gaussian) wave process with an arbitrary power spectrum, and applying non-linear corrections based on second-order hydrodynamics. Inputs to the routine include the first-order spectrum, the water depth, and a set of fixed or moving locations in the along-wave direction at which wave elevation histories are desired. It may thus provide useful input to estimate loads on spatially distributed ocean structures and ships.

The WAVEMAKER package includes a separate driver program, which facilitates input/output and generates several analytical spectral models. Its input is specified in command-line format, similar to that of the TF-POP program for hydrodynamic post-processing also developed in the Stanford RMS Program. Example problems are included to demonstrate the various uses of WAVEMAKER.

In simulation, WAVEMAKER first uses standard frequency domain methods to generate first-order Gaussian histories at each location. For each of these, WAVEMAKER then evaluates the full set of second-order corrections according to hydrodynamic theory. Thus the first-order wave process, with N components at frequencies ω_n, gives rise to a total of N^2 corrections, spread over all sum frequencies $\omega_n + \omega_m$, and to another N^2 corrections over all difference frequencies $\omega_n - \omega_m$.

WAVEMAKER also includes the ability to identify the underlying first-order Gaussian history from a given observed time history. This feature is particularly attractive for use in situations where the second-order non-linearity in the waves is built into the structural response calculations. To avoid double-counting, the input waves should be filtered to remove any second-order nonlinearity. WAVEMAKER takes in an input wave history and identifies its first- and second-order wave components. This identification, an inverse feature to simulation, is based on a Newton-Raphson scheme to solve N simultaneous nonlinear equations to identify the first-order waves which, when run through the second-order wave predictor, matches the observed waves.

Further, WAVEMAKER is capable of using the identified underlying first-order Gaussian wave history to predict consistent first- and second-order wave histories at alternative user-specified locations by individually phase-shifting each of the first-order components and then adding on a second-order correction as described above at the new wave location.

Finally, WAVEMAKER is capable of predicting these consistent first- and second-order histories as they would be observed from a user-specified moving reference point, based on a user-specified motion time-history. This prediction is accomplished by interpolating between a series of predictions at fixed locations.

Chapter 1

Introduction to WAVEMAKER 3.0

WAVEMAKER is a FORTRAN program used to analyze random non-Gaussian ocean wave histories. The program has four distinct capabilities which are described individually in Chapters 2 through 5:

- **Simulate** is used to generate first- and second-order wave histories;
- **Identify** is used to decompose an observed wave history at a single location into its first- and second-order components consistent with second-order wave theory;
- **Predict** is used to construct consistent first- and second-order wave histories at multiple user-specified fixed spatial locations;
- **Translate** is used to predict first- and second-order wave histories as observed from multiple moving reference points based on a single user-specified motion time history.

The program is organized such that a main program driver calls a series of subroutines to read the input data and perform the desired analysis. The various analysis capabilities must be used individually, but generally will be most useful if applied in a sequential manner: the output from identify optimally forms the input to predict and to translate.

This document features one consistent ongoing example, which is used in each chapter to demonstrate the use and capabilities being detailed in that chapter. The input and output files for each application of this example are included with the program on the distribution diskette. Several other examples using measured wave data are also included in the manual to demonstrate the program capabilities. The examples using measured data are not included on the distribution diskette.

This report is largely a compilation of prior RMS reports, with additional descriptive text and some minor corrections to example input (and so output) files to enhance the continuity of the example as it is applied through the four chapters that demonstrate each of WAVEMAKER's capabilities.

In this report, Chapter 2 includes documentation of the simulation capabilities and is largely taken from Report RMS–17. Chapter 3 documents the identification analysis capabilities and is largely taken from Report RMS–22. Chapter 4 documents the prediction capabilities of WAVEMAKER and is largely taken from Report RMS–33. Chapter 5 documents the translation capabilities of WAVEMAKER and is largely taken from Technical note TN–5.

The appendices include sample input and output files for the simulation, identification, prediction, and translation examples.

Chapter 2

Simulation of Second-Order Random Waves

2.1 Introduction

It is common in many ocean engineering problems to seek to simulate a time trace of the wave elevation process, $\eta(t)$, at one or more locations in the along-wave direction. It is most typical to use a *Gaussian* model of $\eta(t)$ for this simulation, which is consistent with linear wave theory. This is due primarily to the ease of simulating Gaussian processes, e.g. with FFT (Fast Fourier Transform) methods for an arbitrary wave spectrum (e.g., Borgman, 1969).

We seek here to demonstrate and facilitate a similar frequency-domain simulation capability for *nonlinear* random waves at a set of spatial locations (e.g., Figure 2.1). These simulations split the wave elevation into a random first-order (linear) wave history, $\eta_1(t)$, and a corresponding nonlinear history $\eta_2(t)$ which includes second-order corrections. FFT techniques are used to generate $\eta_1(t)$ with an arbitrary (first-order) wave spectrum, $S_{\eta_1}(\omega)$. Physical principles are used to generate $\eta_2(t)$ from $\eta_1(t)$, based on second-order perturbation analysis of the underlying nonlinear hydrodynamic problem. Thus if the first-order wave process has N components, at frequencies ω_n, $\eta_2(t)$ includes N^2 second-order corrections, spread over all sum frequencies $\omega_n + \omega_m$, and another N^2 corrections over all difference frequencies $\omega_n - \omega_m$.

Note that these second-order wave models are not novel; they date back at least to the early 1960s (e.g., Longuet-Higgins, 1963). More novel, however, is their recent confirmation with respect to various statistics of field measurements (Marthinsen and Winterstein, 1992; Vinje and Haver, 1994) and wave tank studies of still more

3
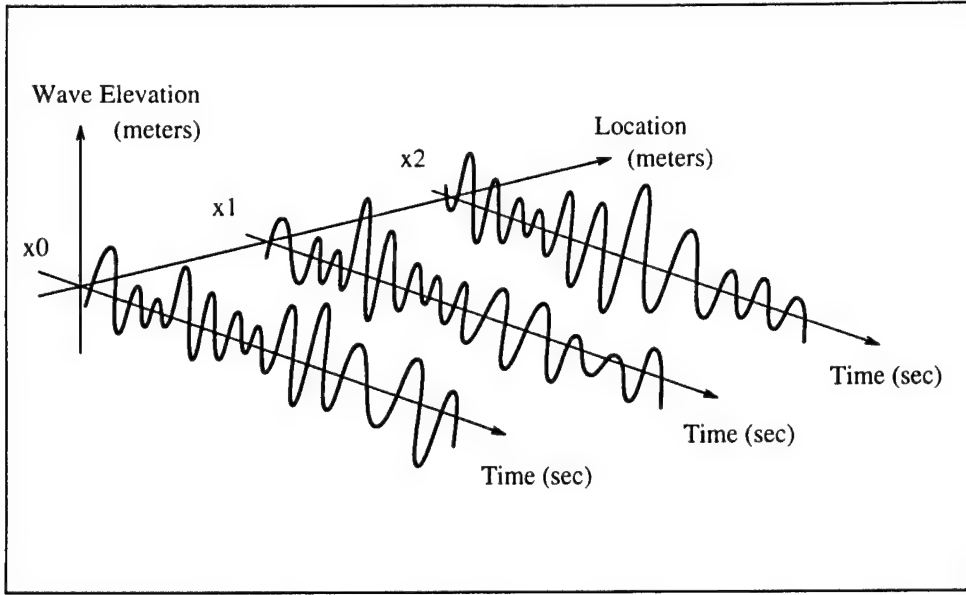
Figure 2.1: Simulated wave time histories at specified spatial locations

severe seas (Winterstein and Jha, 1995). Such second-order wave models also form the basis of state-of-the-art nonlinear diffraction analysis of floating structures (e.g., SWIM, 1995; WAMIT, 1995). Note also that the model of $\eta_2(t)$ used here varies explicitly with water depth, as predicted by second-order theory, to reflect increasing nonlinearity as we proceed to shallower water depths.

2.2   Methodology

2.2.1   Underlying Theory and Assumptions

We first consider $\eta_1(t)$, the first-order wave elevation, at a specific reference location (say $x=0$). For either frequency-domain analysis or time-domain simulation, it is convenient to write $\eta_1(t)$ as a discrete Fourier sum over positive frequencies ω_k:

$$\eta_1(t) = \sum_{k=1}^{N} A_k \cos(\omega_k t + \theta_k) = \operatorname{Re} \sum_{k=1}^{N} A_k e^{i(\omega_k t + \theta_k)} \tag{2.1}$$

To randomize Eq. 2.1, the phases θ_k are taken to be uniformly distributed, mutually independent of each other and of the amplitudes A_k. Furthermore, we assign

random amplitudes A_k with Rayleigh distributions, and mean-square value

$$E[A_k^2] = 2S_\eta(\omega_k)d\omega \equiv \sigma_k^2 \,; \quad d\omega = \omega_k - \omega_{k-1} \tag{2.2}$$

Finally, for purposes of simulation the lowest frequency interval $d\omega$ is governed by the total period T of the simulation:

$$d\omega = \frac{2\pi}{T} \tag{2.3}$$

Together, Eqs. 2.1–2.2 ensure that each of the N frequency components in Eq. 2.1 is itself Gaussian. We also caution against the common use of choosing *deterministic* amplitudes, $A_k = \sigma_k$, particularly when interest lies in preserving higher moments of $\eta_1(t)$—or, similarly, the rms of second-order waves, loads, and responses. Use of deterministic amplitudes can give unconservative estimates; e.g., second-order rms values that are on average too small (Ude, 1994).

The resulting second-order wave at this elevation, $\eta_2(t)$, is calculated from $\eta_1(t)$ as

$$\eta_2(t) = \eta_1(t) + \Delta\eta_2(t) \tag{2.4}$$

in which $\Delta\eta_2(t)$ includes second-order corrections at sums and differences of all wave frequencies:

$$\Delta\eta_2(t) = q\mathrm{Re} \sum_{m=1}^{N} \sum_{n=1}^{N} A_m A_n [H_{mn}^- e^{i[(\omega_m - \omega_n)t + (\theta_m - \theta_n)]} + H_{mn}^+ e^{i[(\omega_m + \omega_n)t + (\theta_m + \theta_n)]}] \tag{2.5}$$

In general, the functions H_{mn}^- and H_{mn}^+ are known as quadratic transfer functions (QTFs), evaluated at the frequency pair ω_m, ω_n. Similar expressions arise in describing loads and responses of floating structures; in this case H_2^- and H_2^+ are calculated numerically from nonlinear diffraction analysis (e.g., WAMIT, 1995). The leading factor q is included in Eq. 2.5 to alert readers to different QTF definitions in the literature: various diffraction analyses use $q=1$ (WAMIT, 1995) or $q=1/2$ (Molin and Chen, 1990).

In predicting motions of floating structures, in view of the relevant natural periods interest commonly lies with either H_2^- (slow-drift) or H_2^+ (springing) but not both. In contrast, in the nonlinear wave problem both sum and difference frequency effects play a potentially significant role. Fortunately, unlike QTF values found numerically from numerical diffraction, closed-form expressions are available for both the sum- and difference-frequency QTFs for second-order waves (e.g., Langley, 1987; Marthinsen

and Winterstein, 1992). Including the effect of a finite water depth d, for example, the sum-frequency QTF can be written as

$$
H_{mn}^+ = \frac{\frac{gk_mk_n}{\omega_m\omega_n} - \frac{1}{2g}(\omega_m^2 + \omega_n^2 + \omega_m\omega_n) + \frac{g}{2}\frac{\omega_m k_n^2 + \omega_m k_n^2}{\omega_m\omega_n(\omega_m+\omega_n)}}{1 - g\frac{k_m+k_n}{(\omega_m+\omega_n)^2}\tanh(k_m+k_n)d}
$$
$$
- \frac{gk_mk_n}{2\omega_m\omega_n} + \frac{1}{2g}(\omega_m^2 + \omega_n^2 + \omega_m\omega_n) \qquad (2.6)
$$

in which the wave numbers k_n are related to the frequencies ω_n by the linear dispersion relation. Note that this QTF definition assumes $q=1/2$ in Eq. 2.5. This is the convention assumed in WAVEMAKER. The corresponding difference-frequency transfer function, H_{mn}^-, is found by replacing ω_n by $-\omega_n$ in Eq. 2.6.

2.2.2   Implementation

On input the simulation method requests the desired number of simulated points, $npts$, and the total duration T to be simulated. To take advantage of discrete FFT (Fast Fourier Transform) techniques, it assumes a regular spacing $dt=T/npts$ between points. Eq. 2.1 is then rewritten as

$$
\eta_1(t) = \sum_{k=1}^{npts/2} A_k\cos(\omega_k t + \theta_k) = \mathrm{Re}\sum_{k=1}^{npts} X_k e^{i\omega_k t} \qquad (2.7)
$$

Here the X_k are complex Fourier coefficients. The lower half of these directly reflect both the random amplitude A_k and phase θ_k at frequency $\omega_k=k\cdot d\omega$:

$$
X_k = \frac{1}{2}A_k e^{i\theta_k}; \quad k = 1...npts/2 \qquad (2.8)
$$

The upper half are in turn taken as the complex conjugates (the symbol "*") of the lower half:

$$
X_{npts-k} = X_k^*; \quad k = 1...npts/2 \qquad (2.9)
$$

This reflects that unique information is contained only the lower-half frequencies; indeed, any information in the upper half frequencies (above the Nyquist) is obscured by aliasing.

Thus, the first-order wave process is generated by assigning random A_k and θ_k, defining X_k from Eqs. 2.8–2.9, and finally taking the inverse Fourier transform to recover the discretized time history $\eta_1(t_j)$. To see this, note that since $dt\cdot d\omega=2\pi/npts$,

Eq. 2.7 can be evaluated at $t=t_j$ to give

$$\eta_1(t_j) = \text{Re} \sum_{k=1}^{npts} X_k e^{2\pi ijk/npts} \qquad (2.10)$$

This is precisely the definition of the discrete FFT.

If a maximum first-order frequency, *omgmax*, is specified in the input file, then a value *nmax* is calculated as *nmax=omgmax/dω*. If k in Eq. 2.8 is greater than *nmax* then the complex Fourier coefficient, X_k, is set equal to zero. Note that the maximum frequency at which energy will be assigned by the sum terms in the second-order transfer function Eq. 2.5 and Eq. 2.11 (below) then becomes $2 \cdot$ *omgmax*. The energy associated with those first-order frequencies that are set equal to zero is *not* assigned elswhere in the spectrum, i.e. the total energy in the spectrum is slightly decreased.

As a minor technical issue, note that the conjugate symmetry here ensures that the "Real Part" operation in Eq. 2.1 is superfluous; i.e., no imaginary component is generated. Also, to conform with the FFT routine used the array of X_k values is shifted by one index: i.e., X_1 corresponds to the frequency zero (the steady term, defined as zero), X_2 to frequency $\omega_1=d\omega$, X_3 to frequency $\omega_2=2 \cdot d\omega$, and so forth.

The second-order correction is generated similarly. Starting with Eq. 2.5, substituting $q=1/2$, $N=npts/2$ (Eq. 2.7) and X_k from Eq. 2.8:

$$\Delta\eta_2(t) = 2\text{Re} \sum_{m=1}^{npts/2} \sum_{n=1}^{npts/2} X_m X_n H_{mn}^+ e^{i(\omega_m+\omega_n)t} + X_m X_n^* H_{mn}^- e^{i(\omega_m-\omega_n)t} \qquad (2.11)$$

The leading factor reflects the product of $q=1/2$ and a net factor of 4 (since $A_n A_m$ is $4|X_m X_n|$). The program then seeks to rewrite both the sum and difference frequency contributions in a Fourier sum analogous to Eq. 2.10. For example, the sum-frequency is assumed of the form

$$\Delta\eta_2^+(t_j) = \text{Re} \sum_{k=1}^{npts} Y_k e^{2\pi ijk/npts} \qquad (2.12)$$

The output Fourier coefficients, Y_k, are evaluated by equating Eqs. 2.11 and 2.12. This implies a sum over all wave frequency pairs (ω_m, ω_n) in Eq. 2.11 that give rise to output sum frequency ω_k. The difference frequency Fourier coefficients are constructed in a similar way, and added on the sum frequency Y_k coefficients. Once these combined Y_k coefficients are found a (one-dimensional) inverse FFT is performed to recover the second-order time history.

2.2.3   Multiple Spatial Locations

The linear dispersion relation can be used to generalize Eq. 2.4, which generates a first-order wave at reference location $x=0$, to any other spatial location x in the along-wave direction. The linear dispersion relation is first used to find the wave number k_n associated with each frequency ω_n in Eq. 2.4. The modified first-order simulation then merely replaces $\omega_n t + \theta_n$ in Eq. 2.4 by $\omega_n t - k_n x + \theta_n$. Equivalently, the original phases θ_n are first shifted to $\theta_n - k_n x$ before applying Eq. 2.8 to define the wave Fourier amplitude X_n. These appropriately modified X_n are also used in Eq. 2.11 to find the corresponding second-order correction at this new location.

2.3   Input Specification

This section describes the various inputs required by the program and the syntax of the input to the driver routine for WAVEMAKER. This input is provided in the following format:

keyword *args*
where keyword is a reserved word and *args* are its arguments. A typical input file is in the following format:

```
# Typical input file: syntax description

simulate duration npts seed
depth value
psd psdtype psd_parameters
define varlimit value
define gravity value
define omgmax value
write history filename1 filename2
write statistics filename3 filename4
location nloc
value1
value2
   ⋮
valuenloc
```

Each of these lines in a typical input file is explained below:

Typical input file: syntax description

Any line beginning with a "#" is treated as a comment line in the input file and is ignored by the program. Blank lines are also ignored by the program.

simulate *duration npts seed*

The keyword **simulate** indicates to the program that the following three arguments in sequence are:

- *duration*: Total desired duration (in seconds) of each of the simulated wave histories (a real number).
- *npts*: Number of points required in each of the simulated wave histories (an integer).
- *seed*: A real number (131071.0 for example) for generation of random numbers. The seed may be changed by the user in order to generate a different set of wave histories. The number should be between 1.0 and 2^{31}.

The resulting dt (time step) in the simulated histories is *duration/npts*. The *mxgrd* variable in the driver program controls the maximum number of points allowed (Maximum *npts* = $2 \times mxgrd$) in a simulation. The released driver program has *mxgrd* = 4096, so that specified *npts* can to up to 8192. The user may increase or decrease *mxgrd* to suit his/her needs.

depth *value*

This line specifies the water depth at the site of interest. The keyword is **depth** and *value* is a real number indicating the water depth. The units (meters, feet, etc.) of this value should be consistent with the units of other input parameters.

psd *psdtype psd_parameters*

This line specifies the spectrum type to be used. The keyword is **psd** followed by its arguments. *psdtype* may be one of the following: jonswap, bimodal, or boxcar. If any other word is specified for *psdtype*, then it indicates to the program that an input spectrum is specified in a file whose name is same as the word specified in place of *psdtype*. More details regarding this input specification are given in the following subsection.

define varlimit *value* [optional command]
If included, this line defines a constant **varlimit** whose value is a real number (between 0.0 and 1.0) equal to *value*. A warning is issued by the simulation routine, if the estimated second-order power above Nyquist frequency is more than *value* times the first-order power. If this line is not provided in the input file, then a default value of 0.01 is assigned to **varlimit**.

define gravity *value* [optional command]
If included, this line specifies the acceleration due to gravity in consistent units. *value*, a real number, is assigned to **gravity**. If not included, a default value of 9.807 meters/sec^2 is assumed.

define omgmax *value* [optional command] If included, this line specifies the maximum frequency, in radians per second, for which energy will be assigned in the specified wave spectrum. Specifically, if k in Eq. 2.8 is greater than *omgmax/dω*, then the Fourier component X_k is set equal to zero. The specified spectrum is otherwise unchanged.

write history *filename1 filename2* [optional command]
If included, this line specifies the files which contain the simulated time histories at each location. The keyword is **write history**. The file *filename1* contains the underlying first-order (Gaussian) histories, and *filename2* contains the corresponding second-order wave histories. The format of the output is presented in the following section. If this line is not provided in the input file then default names of **gauss.sim** (**Gauss**ian **sim**ulation) and **ngauss.sim** are assigned to the output first- and second-order histories, respectively.

write statistics *filename3 filename4* [optional command]
This line specifies the files to which the statistics (mean, standard deviation, skewness, kurtosis, minimum, and maximum value) of the simulated histories should be written. The keyword is **write statistics**. The statistics for the simulated first-order histories at each spatial location is written out in *filename3* and the statistics for the total second-order histories are written in *filename4*. If this line is not provided in the input file then default names of **gauss.sst** (**gauss**ian simulation **st**atistics) and **ngauss.sst** are assigned to the output files for the first- and second-order history statistics, respectively.

location *nloc*

This line specifies the number of spatial locations at which both the first- and total second-order wave histories should be simulated in time. The keyword is **location**. *nloc* (an integer) specifies the number of locations (maximum location allowed is 50). *value1*, *value2*, ..., *valuenloc* are the spatial values (real numbers) in consistent units. The number of values should be equal to *nloc*. The user is forewarned that the specification of the spatial locations should be at the end of all other inputs required.

2.3.1   Wave Spectrum Specification

The input wave spectrum can be specified in various ways: spectrum values in an input file, or spectrum type with related parameters from a default library.

psd *filename*
This specifies that the input spectrum be read from an input file named *filename*. The format of this file is two free-formatted values per line, specifying a natural frequency in radians per second and the one-sided PSD (power spectral density) ordinate for that frequency in consistent units of squared amplitude (feet^2, meters^2, etc.) per rad/sec. Lines beginning with a "#" are regarded as comments and ignored. The input spectrum may be specified on an irregular mesh. This is internally linearly interpolated to a spectrum on a regular mesh specified by *duration* and *npts*. The spectral ordinates below the minimum frequency and above the maximum frequency specified in the irregular mesh are assumed to be zero.

Spectral models from the library are called upon using any one of the following reserved names followed by their parameters (that are real numbers):

psd jonswap H_s T_p γ
psd bimodal H_s T_p
psd boxcar σ_η ω_{lo} ω_{hi}

The keyword **jonswap** invokes a JONSWAP spectrum parameterized by the significant wave height H_s (defined as four times the standard deviation of the wave elevation process), spectral peak period T_p (in seconds), and the peakedness factor γ.

The keyword **bimodal** invokes a spectral model proposed by Torsethaugen (Bitner-Gregerson and Haver, 1991). This subdivides the H_s–T_p scattergram into three regions, and assigns bimodal spectral shapes in several of these regions. Therefore, the only input required for this **bimodal** option is H_s and T_p.

Finally, the keyword **boxcar** invokes a simple band-limited white-noise model of the first-order wave spectrum. Its parameters are the rms σ_η, lower cutoff frequency ω_{lo}, and upper cutoff frequency ω_{hi} of the first-order wave spectrum. As in other cases (e.g., the user-defined spectrum at various frequencies), the frequencies ω_{lo}, and ω_{hi} here are assumed to be in units of rad/sec. Note also that non-zero values of the PSD at zero frequency are not allowed in either file input or library model selections.

2.4   Output Format

A total of four output files are produced by the driver program. Two output files contain the time histories: one for the underlying first-order wave histories and the other for the total second-order histories. The other two output files contain wave statistics: first four moments, minimum and maximum. Again, results for the first- and second-order wave histories are separated into two files.

2.4.1   Time History Output

As noted in the previous section, by default the first- and second-order histories are written to the files **gauss.sim** and **ngauss.sim**. Other choices of output filenames can be specified by the optional **write history** command. The format of this output depends on the number of spatial locations specified. If the number of locations (*nloc*) is less than or equal to 8 then the output is in **Format1** otherwise the output is in **Format2**. Both of these formats write out 3 header lines beginning with a "#" sign. These are to be treated as comment lines in the output file. In order to explain the two format styles, say that the spatial locations specified are x1, x2, x3, ..., xnloc.

Format1 outputs data in *nloc*+1 columns. The length of each column is equal to the number of points desired in each simulation. The first column contains the time increments in seconds going from 0 to T with dt $= T/npts$. Columns 2 through *nloc*+1 contain the simulated history values at the specified locations x1, x2, x3, ...xnloc, respectively. Thus, column 2 contains the wave elevation at location x1, column 3 contains wave elevation at location x2, and so on.

Format2 is for handling *nloc* greater than 8. The output begins with the time increment Ti in seconds on a line by itself. The time history values for the specified spatial locations at time Ti are written in the next line onwards, in sets of 10. So if 9 locations were specified (i.e., *nloc* = 9) then the time increment is printed on a line by itself followed by a line containing 9 time history values at that time increment. The next line contains the next time increment followed by another set of 9 values, and so on. If, on the other hand say 28 locations were specified, then a time increment is written on a line followed by 28 time history values (corresponding to 28 locations at that time increment) in the next 3 lines. The first line of the 3 lines contains 10 time history values for the first 10 locations specified. The next line contains 10 history values for locations 11 through 20 and the following line which is the third line of the set will contain only 8 history values for location 21 through 28.

2.4.2   Wave Statistics Output

The statistics of the simulated histories are also estimated by the driver program. These statistics include the mean, standard deviation, skewness, kurtosis, minimum, and maximum. As noted in the previous section, first- and second-order simulation results are written by default to the files **gauss.sst** and **ngauss.sst**, respectively. The optional command **write statistics** can alter this choice of output filenames.

The output format in both of these files begins with 2 header lines, each of which begins with a "#" sign. The output is in seven columns. The first column specifies the spatial location. The following six columns contain statistics of the wave history at the spatial location specified in column 1. Columns 2 through 7 contain, the mean, standard deviation, skewness, kurtosis, minimum, , and maximum value in that order.

The next section presents some sample output files, to illustrate use of the WAVEMAKER routine.

2.5   Example

In this section, we present a sample problem (copies of the output files are included in Appendix A and enclosed on the distribution diskette).

To illustrate, consider a simulation which samples the wave process at regular intervals of length dt=0.5 [sec] over a total duration of T=2048 [sec]; i.e., $npts$=4096 points. We assume here the first-order wave spectrum to be of JONSWAP form, with

$H_s = 12$ [m], $T_p = 14$ [sec] and a peakedness factor $\gamma = 3.3$. We further seek to generate wave histories at 2 spatial locations: 0 and 60 [m].

The example presented here differs from that presented in Report RMS–33 in that a high frequency cut-off at *omgmax*$= \pi/2 = 1.57$ radians per second is applied. This frequency corresponds to $1/4$ of the Nyquist frequency for this example ($dt = 0.5$). The high-frequency cut-off is implemented on the complex Fourier coefficients, X_k, in Eq. 2.8 prior to generation of the first-order wave components. Second-order energy (Eq. 2.5) will be assigned at frequencies as high as $2 \cdot$ *omgmax*, or 3.14 radians per second.

The input file for simulating waves using **WAVEMAKER** is:

```
# Gaussian and Nongaussian Wave Input File

simulate 2048.0 4096 8123872.0
depth 70.0
psd jonswap 12.0 14.0 3.3
write history gauss.sim ngauss.sim
write statistics gauss.sst ngauss.sst
define varlimit 0.01
define gravity 9.807
define omgmax 1.57
location 2
0.0
60.0
```

Alternatively, if we intend to use the default definitions in the program then our input file could be (this will produce the same output as the extended version of the input file):

```
# Gaussian and Nongaussian Wave Input File
# (Alternative format)

simulate 2048.0 4096 8123872.0
depth 70.0
psd jonswap 12.0 14.0 3.3
define omgmax 1.57
location 2
0.0
60.0
```

The output files created are: **gauss.sim**, **ngauss.sim**, **gauss.sst**, and **ngauss.sst**. The contents of these are listed in the following table:

Output File	Contents
gauss.sim	First-Order Time History
ngauss.sim	Second-Order Time History
gauss.sst	First-Order History Statistics
ngauss.sst	Second-Order History Statistics

Portions of each of these output files are given in Appendix A.

Figures 2.2 and 2.3 show a comparison of the simulated histories. Figure 2.2 compares the simulated first- and second-order wave time histories at the same location (the first of the two requested, defined arbitrarily as $x=0$ [m]). The file **ngauss.sst** includes the estimated skewness of the second-order waves. At this location the second-order wave history has a skewness of about 0.2. This positive skewness (compared to zero skewness of Gaussian waves) indicates the systematic nonlinear effects. This also gives rise to an asymmetry between peaks and troughs; in particular, the extreme wave crest in the second-order simulation systematically exceeds the corresponding extreme trough in absolute value. This tendency may be significant for potential deck impact problems, particularly in older jacket structures with relatively low deck levels.

Figure 2.3 compares the second-order wave histories predicted by simulation at the two spatial locations, separated by 60 [m]. The phase lag at these two locations is evident in the plot. It can be seen that a crest height at 0 [m] does not necessarily imply a crest of the sar ⁘ height at 60 [m].

Figure 2.4 shows the results of this example with and without imposition of the high frequency cut-off of *omgmax*=1.57 radians per second.
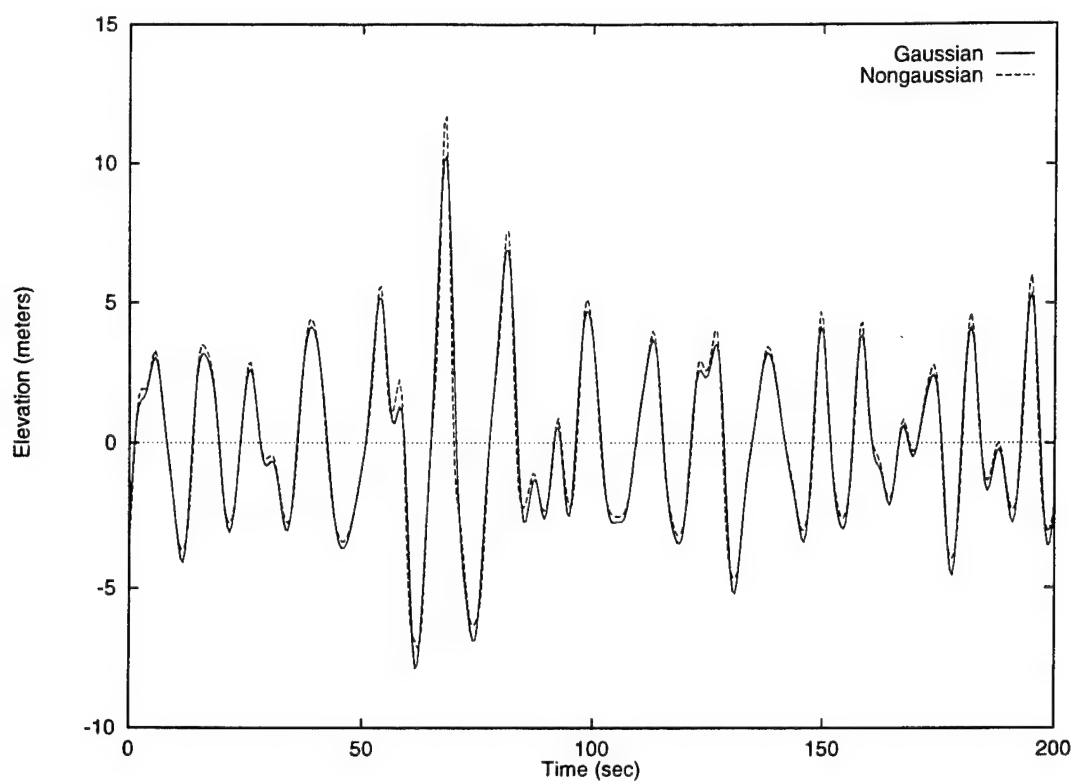
Figure 2.2: Simulated first- and second-order wave histories at location 0.0
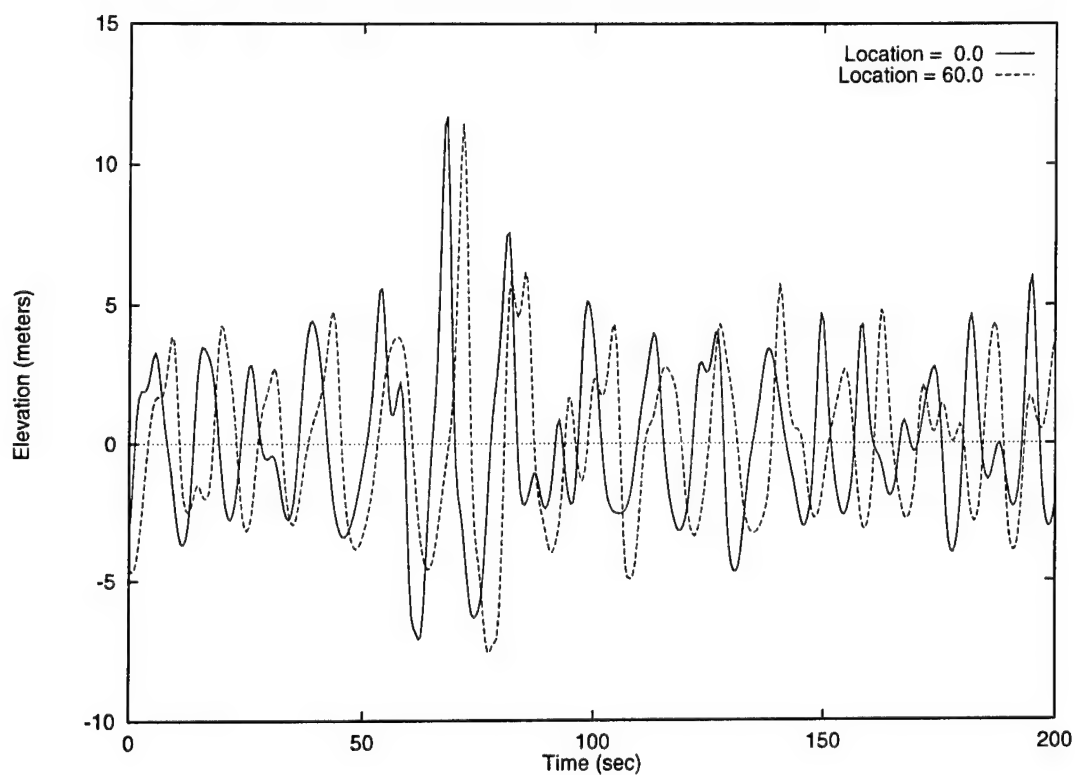
Figure 2.3: Simulated second-order wave histories at locations 0.0 and 60.0
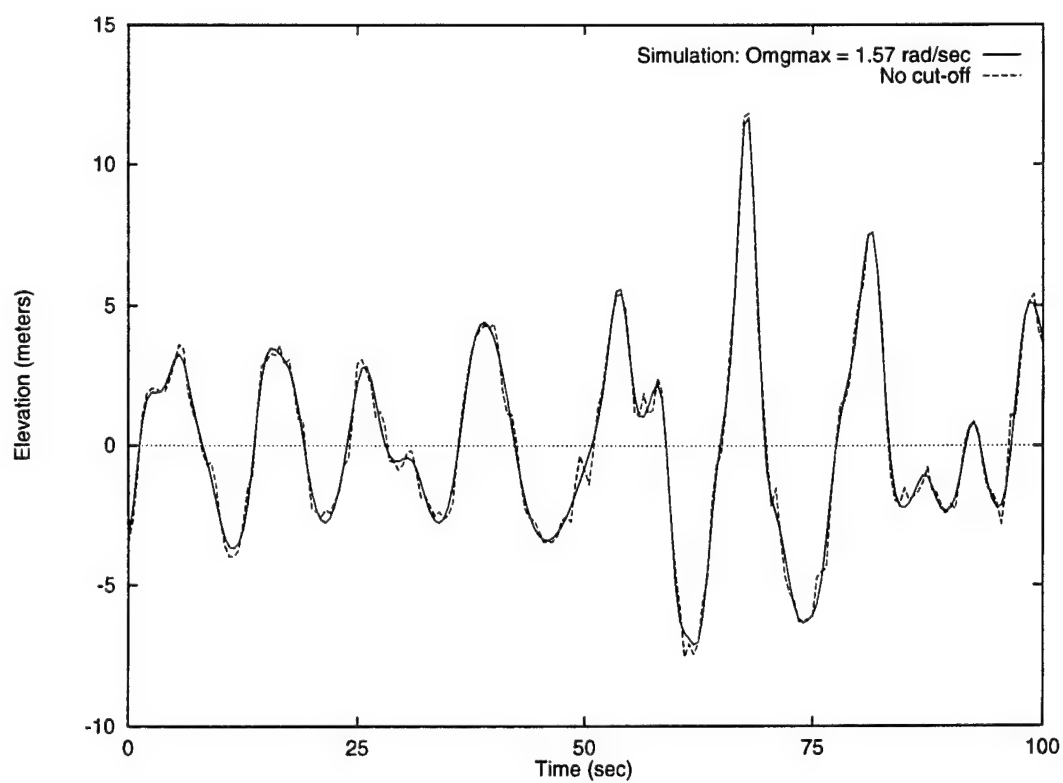
Figure 2.4: Simulated second-order wave histories at location 0.0 with and without high frequency cut-off

Chapter 3

Identification of First-Order Waves

3.1   Introduction

In ocean engineering practice it is common to assume the waves to be Gaussian and any nonlinearity in the waves is embedded in the structural response analysis (e.g., WAMIT, 1995). It has been shown in Winterstein and Jha, 1995 that observed time histories generally contain nonlinearities, it is thus imperative to remove any second-order effects in the incident waves so that we do not double-count these effects in the resulting response estimation. Recent studies (Ude and Winterstein, 1996) have demonstrated the impact of double-counting such second-order effects on various structural response characteristics.

The methodology to identify the underlying first-order waves is to seek the implied first-order wave history which, when run through the second-order wave predictor, yields an incident wave that agrees with the target observed history at each time point. This identification is performed using a Newton-Raphson scheme to achieve simultaneous convergence at each complex Fourier component. If the observed history has N components, we iteratively solve N simultaneous nonlinear equations to identify the first-order components.

Due to computer memory limitations, the identification of the first-order history is performed on short contiguous windows of the observed history. This window size (*mlen*) can be made equal to the observed history length in WAVEMAKER if the computer has sufficient RAM and swap space.

3.2   Methodology

The idea here is to identify the implied first-order history $\eta_1(t)$ (of an observed history $\eta_{\text{obs}}(t)$) which, when run through the second-order predictor, yields an incident wave that agrees with $\eta_{\text{obs}}(t)$. In the first-order wave process $\eta_1(t)$ (see Eq. 2.7), written as a Fourier sum of N frequencies,

$$\eta_1(t) = \sum_{k=1}^{N/2} A_k \cos(\omega_k t + \theta_k) = \sum_{k=1}^{N} X_k e^{i\omega_k t} \tag{3.1}$$

we need to identify only the lower half X_k components, since the upper half values are complex conjugates of the lower half. Let us denote $X_k = U_k + iV_k$, where U_k, V_k are the real amd imaginary parts of the complex Fourier component X_k, respectively.

The predicted second-order wave process (see Eq. 2.11) as evaluated from the QTFs is

$$\Delta\eta_2(t) = 2\text{Re} \sum_{m=1}^{N/2} \sum_{n=1}^{N/2} X_m X_n H_{mn}^+ e^{i(\omega_m+\omega_n)t} + X_m X_n^* H_{mn}^- e^{i(\omega_m-\omega_n)t} \tag{3.2}$$

This may be rewritten in the form of a Fourier sum as

$$\Delta\eta_2(t) = \sum_{k=1}^{N} Y_k e^{i\omega_k t} \tag{3.3}$$

where $Y_k = Y_k^+ + Y_k^-$ are the combined sum and difference frequency components. Here again, Y_k possesses conjugate symmetry so that only the lower half contains unique information. Y_k^+ can be shown to be

$$\begin{aligned} Y_k^+ &= \sum_{m+n,k} X_m X_n H_{mn}^+ \\ &= \sum_{m+n,k} \left[(U_m U_n - V_m V_n) + i(V_m U_n + U_m V_n) \right] H_{mn}^+ \end{aligned} \tag{3.4}$$

where the summation symbol indicates a double summation

$$\sum_{m+n,k} = \sum_{m=1}^{N/2} \sum_{n=1}^{N/2} \quad \text{such that } \omega_m + \omega_n = \omega_k \tag{3.5}$$

and

$$\begin{aligned} Y_k^- &= \sum_{m-n,k} X_m X_n^* H_{mn}^- \\ &= \sum_{m-n,k} \left[(U_m U_n + V_m V_n) + i(V_m U_n - U_m V_n) \right] H_{mn}^- \end{aligned} \tag{3.6}$$

where

$$\sum_{m-n,k} = \sum_{m=1}^{N/2} \sum_{n=1}^{N/2} \quad \text{such that } |\omega_m - \omega_n| = \omega_k \tag{3.7}$$

The combined predicted wave process is

$$\eta_{\text{pred}}(t) = \eta_1(t) + \Delta\eta_2(t) \tag{3.8}$$

The identification scheme strives to simultaneously match $\eta_{\text{pred}}(t)$ to the observed wave history $\eta_{\text{obs}}(t)$ at every value of t. Alternatively, we can perform the identification in the frequency domain and strive to simultaneously match the predicted Fourier components to the observed Fourier components at all frequencies.

$\eta_{\text{obs}}(t)$ can be represented in the frequency domain as

$$\eta_{\text{obs}}(t) = \sum_{k=1}^{N} Z_k e^{i\omega_k t} \tag{3.9}$$

where Z_k's also possess conjugate symmetry. If the first-order components are identified exactly, from Eq.s 3.1, 3.3 and 3.9 we will have

$$Z_k = X_k + Y_k \quad ; \quad \text{for all } k = 1 \ldots N/2 \tag{3.10}$$

Note that the upper half values can be obtained from conjugate symmetry of the lower half values. In the Newton-Raphson identification scheme we will try to simultaneously minimize $X_k + Y_k - Z_k$; for $k = 1 \ldots N/2$ to achieve convergence. Now, this scheme requires a Jacobian of $X_k + Y_k - Z_k$ with respect to the unknowns X_k–such a complex differentiation will lead to numerical discontinuities so we will minimize an equivalent real function $\sqrt{\sum_1^N f_k^2/N}$ instead, where for $k = 1 \ldots N/2$

$$\begin{aligned} f_k &= \text{Re}(X_k + Y_k - Z_k) \\ f_{k+N/2} &= \text{Im}(X_k + Y_k - Z_k) \end{aligned} \tag{3.11}$$

The identification of the lower half X_k values requires a simultaneous solution of the nonlinear equations in 3.11 such that $f_k \to 0$ for all $k = 1 \ldots N$, or alternately $\sqrt{\sum_1^N f_k^2/N} \to 0$. We will formulate the Newton-Raphson scheme in vector form as

$$\mathbf{f} = \begin{bmatrix} \text{Re}\mathbf{X} \\ \hline \text{Im}\mathbf{X} \end{bmatrix} + \begin{bmatrix} \text{Re}\mathbf{Y} \\ \hline \text{Im}\mathbf{Y} \end{bmatrix} - \begin{bmatrix} \text{Re}\mathbf{Z} \\ \hline \text{Im}\mathbf{Z} \end{bmatrix} \tag{3.12}$$

where bold face letters denote vectors, and vectors $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ contain the complex Fourier components X_k, Y_k, Z_k, $k = 1 \ldots N/2$, respectively. Here, $\begin{bmatrix} \frac{\text{Re}\mathbf{X}}{\text{Im}\mathbf{X}} \end{bmatrix}$ is a vector containing the real part of $\mathbf{X}$ in the upper half and the imaginary part of $\mathbf{X}$ in the lower half.

Let us denote

$$
\mathbf{A} \;=\; \left[\frac{\mathrm{Re}\mathbf{X}}{\mathrm{Im}\mathbf{X}} \right] = \left[\frac{\mathbf{U}}{\mathbf{V}} \right]
$$

$$
\mathbf{B} \;=\; \left[\frac{\mathrm{Re}\mathbf{Y}}{\mathrm{Im}\mathbf{Y}} \right] \tag{3.13}
$$

$$
\mathbf{C} \;=\; \left[\frac{\mathrm{Re}\mathbf{Z}}{\mathrm{Im}\mathbf{Z}} \right]
$$

Note that the vector $\mathbf{A}$, of length N, is constructed such that lower half values are the real parts of X_k; $k = 1 \ldots N/2$ and the upper half is the imaginary part of X_k; $k = 1 \ldots N/2$. Similarly, $\mathbf{B}$ and $\mathbf{C}$, each of length N, contain real and imaginary parts of the lower half of the second-order correction and the observed Fourier components, respectively. The elements of $\mathbf{A}$ and $\mathbf{B}$ are denoted by a_l and b_k, respectively, where $l, k = 1 \ldots N$. The objective function in vector notation now is

$$
\mathbf{f}(\mathbf{A}) = \mathbf{A} + \mathbf{B} - \mathbf{C} \tag{3.14}
$$

A first-order Taylor approximation of $\mathbf{f}(\mathbf{A})$ about a given $\mathbf{A}^{(0)}$ is

$$
\mathbf{f}\left(\mathbf{A}\right) = \mathbf{f}\left(\mathbf{A}^{(0)}\right) + [J]\left(\mathbf{A} - \mathbf{A}^{(0)}\right) \tag{3.15}
$$

where $[J]$ is a $N{\times}N$ Jacobian matrix denoting the derivatives of the elements f_k in vector $\mathbf{f}(\mathbf{A})$ with respect to each of the unknowns a_l in $\mathbf{A}$ where $k, l = 1 \ldots N$. The Newton-Raphson scheme at iteration $p + 1$ is then formulated as

$$
\mathbf{A}^{(p+1)} = \mathbf{A}^{(p)} + \mathbf{h} \tag{3.16}
$$

where $\mathbf{h}$, a vector of length N, is found from a Cholesky decomposition followed by a back-substitution scheme from

$$
[J]\mathbf{h} = -\mathbf{f}\left(\mathbf{A}^{(p)}\right) \tag{3.17}
$$

It can be easily shown from Eq. 3.14 that the entries $J_{k,l}$ of the matrix $[J]$ are

$$
J_{k,l} = \frac{\partial f_k}{\partial a_l} = \delta_{kl} + \frac{\partial b_k}{\partial a_l} \tag{3.18}
$$

where $\partial b_k / \partial a_l$ indicates the partial derivative of b_k with respect to a_l, and

$$
\delta_{kl} = \begin{cases} 1 & \text{if } k = l \\ 0 & \text{otherwise} \end{cases} \tag{3.19}
$$

To find $\partial b_k / \partial a_l$, recall from notation in 3.13

$$
\begin{aligned}
b_k &= \mathrm{Re}Y_k \quad \text{and} \quad b_{k+N/2} = \mathrm{Im}Y_k \quad \text{for } k = 1 \ldots N/2 \\
a_l &= \mathrm{Im}X_l = U_l \quad \text{and} \quad a_{l+N/2} = \mathrm{Im}X_l = V_l \quad \text{for } l = 1 \ldots N/2
\end{aligned}
$$

so that from Eq.s 3.4 and 3.6 we have

$$\frac{\partial \text{Re}Y_k}{\partial U_l} = \sum_{m+n,k} (U_n \delta_{ml} + U_m \delta_{nl}) H_{mn}^+ + \sum_{m-n,k} (U_n \delta_{ml} + U_m \delta_{nl}) H_{mn}^-$$

$$\frac{\partial \text{Re}Y_k}{\partial V_l} = \sum_{m+n,k} -(V_n \delta_{ml} + V_m \delta_{nl}) H_{mn}^+ + \sum_{m-n,k} (V_n \delta_{ml} + V_m \delta_{nl}) H_{mn}^- \quad (3.20)$$

$$\frac{\partial \text{Im}Y_k}{\partial U_l} = \sum_{m+n,k} (V_m \delta_{nl} + V_n \delta_{ml}) H_{mn}^+ + \sum_{m-n,k} (V_m \delta_{nl} - V_n \delta_{ml}) H_{mn}^-$$

$$\frac{\partial \text{Im}Y_k}{\partial V_l} = \sum_{m+n,k} (U_n \delta_{ml} + U_m \delta_{nl}) H_{mn}^+ + \sum_{m-n,k} (U_n \delta_{ml} - U_m \delta_{nl}) H_{mn}^-$$

Schematically,

$$[J] = [I] + \begin{bmatrix} \dfrac{\partial \text{Re}Y_k}{\partial U_l} & \dfrac{\partial \text{Re}Y_k}{\partial V_l} \\ \dfrac{\partial \text{Im}Y_k}{\partial U_l} & \dfrac{\partial \text{Im}Y_k}{\partial V_l} \end{bmatrix} \quad (3.21)$$

where $[I]$ is the identity matrix.

3.2.1 Ramp

The *identify* option breaks the input wave time history into manageably sized contiguous windows to reduce computational intensity and computer memory demands. Unfortunately, the waves generally are not continuous in slope or offset between opposite ends of each of these conveniently sized data windows. This discontinuity requires special treatment to avoid difficulties in the application of fast Fourier transform (FFT) techniques because of the "wrap-around" effect inherent to the FFT process. In computationally intensive applications, such as the *identify* option, it is important that the number of points in each of these windows be specified as an even power of 2 to take advantage of the computational efficiencies of the FFT. The need to control the exact number of points in each data window precludes use of the method used in the *predict* option described in Section 4.4.

If an end discontinuity is present, the FFT attempts to duplicate the time-domain discontinuity by introduction of large high frequency components to the frequency domain characterization of the wave profile. Any numerical noise introduced by end discontinuities is disbursed throughout the time history when the inverse fast fourier transform (IFFT) is taken to return the wave history to the time domain: The "noise" created by an endpoint discontinuity is spread over the entire data window.

The discontinuities, and so the high frequency "noise," can be suppressed by

prepending and appending a ramping sequence to the wave history in each window. Implementation of a ramping sequence was an enhancement originally added to WAVEMAKER 3.0 and was left unchanged in WAVEMAKER 3.1; it has been further revised in this WAVEMAKER 3.2.

Ramps As Applied in WAVEMAKER 3.1

In WAVEMAKER 3.1, the ramp function was a sinusoid within a decaying (or expanding) envelope linearly raised or collapsed to the mean of the data, usually near zero. The modulated sine function within the ramping envelope was described by:

$$R(t + t_d) = \bar{x} + \frac{t + t_d}{t_d}[x_0 \cos(\omega t) + \frac{v_0}{\omega} \sin(\omega t)] \quad (-t_d < t < 0) \tag{3.22}$$

in which $\bar{x}$ is the mean of the observed input time series and $x_0 = x(0) - \bar{x}$ and $v_0 = (x(1) - x(0))/dt - x_0/t_d$ are the initial value and modified velocity of the zero mean input x. dt is the sampling time interval and t_d is the additional time by which each end of the data window is to be expanded. The frequency ω of the ramp is estimated as the average frequency of the first and last ten cycles of the input time series. A sample window including the generated ramp points is shown in Figure 3.1.

Theoretical problems have been identified with this methodolgy. First, use of the Fourier transform implies an assumption of a stationary process. Ramping down to zero at either end of the time history creates a "pulse," which clearly violates the assumption of stationarity. Second, the previous implementation allowed the possibility of two types of discontinuities: a phase discontinuity and a frequency discontinuity.

The phase discontinuity is not obvious in a plot of a "ramped" time history because the ramps end at zero amplitude, and so there is no offset or slope discontinuity between the endpoints. There also *appears* to be no phase discontinuity between the endpoints, when in fact there can be a phase difference which even at zero amplitude causes introduction of high frequency energy by the FFT process.

A frequency discontinuity can also be introduced in WAVEMAKER 3.1: if the average frequency at the beginning end of the wave data differs from the average frequency at the other end, the two ramps would have been implemented at these differing wave frequencies. Again, this difference can cause introduction of high frequency energy at the FFT.

The current version, WAVEMAKER 3.2, includes a new implementation of the ramp
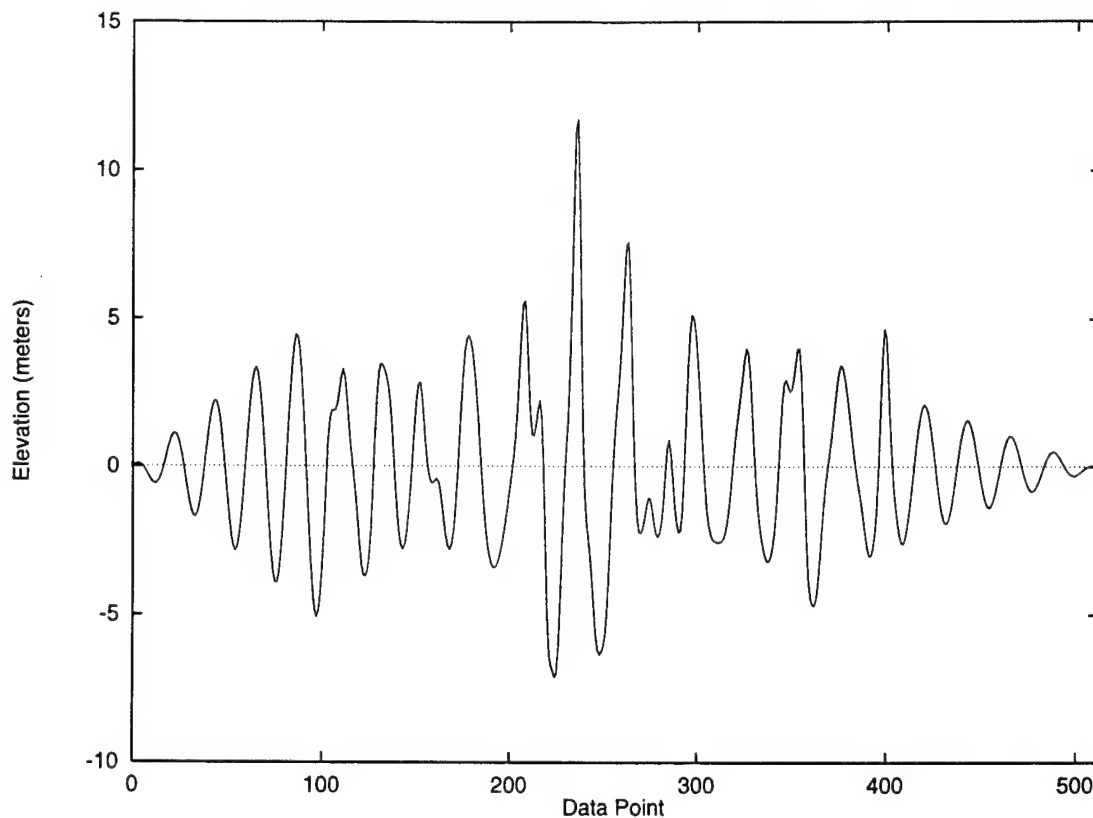
Figure 3.1: Single Data Window including Ramps as generated in Previous WAVE-MAKER 3.1. The first and last 100 data points are artificially generated ramps.

concept which corrects the problems associated with the previous implementation.

Ramps As Applied in WAVEMAKER 3.2

The ramp function applied in `WAVEMAKER 3.2` is again a sinusoid within a linearly decaying (or expanding) envelope. In this version, however, it is linearly raised or collapsed at each end of the expanded window to the mean amplitude of the peaks of the input data. This mean peak amplitude, $\bar{A}$, is robustly calculated from the standard deviation of the process and an assumed Raleigh distribution: $\bar{A} = \sigma\sqrt{\pi/2}$, where σ refers to the standard deviation of the wave process. The ramp function, starting at the left ($-t_d$) end of the expanded data window, is now:

$$R(t + t_d) = \bar{x} + [\bar{A} - \frac{t}{t_d}(\bar{A} - 1)][x_0\cos(\omega t) + \frac{v_0}{\omega}\sin(\omega t)] \quad (-t_d < t < 0) \qquad (3.23)$$

in which $\bar{x}$ is the mean of the observed input time series and t is the time for which a new value of x is to be defined. $x_0 = x(0) - \bar{x}$ and $v_0 = (x(1) - x(0))/dt - x_0/t_d$ are the initial value and modified velocity (slope) of the input x where $x(0)$ and $x(1)$ are the first and second input data points in the window. is the sampling time interval and t_d is the additional time by which each end of the data window is to be expanded.

The frequency ω of the ramp is first estimated as the average frequency of the complete input time series in the present window and then adjusted such that there are *exactly* θ plus an integer number of sinusoids in the ramp. θ is that portion of a sinusoid needed to "complete" the first wave cycle in the window such that it begins as a sinusoid at $\pi/2$:

$$\theta = \arctan \frac{v_o}{x_0} + \pi \quad (0 < \theta < 2\pi). \tag{3.24}$$

Continuity between the ends of the window is assured; each end of the augmented window begins at a local maximum of amplitude, $\bar{A}$, and each end has nearly the same frequency, ω. A sample window including the generated ramp points is shown in Figure 3.2.

In the event that one end of the ramp would be more than 40% larger than the largest peak in the data window, the ramp envelope is restricted to 1.4 times that maximum peak. The reduced envelope is implemented by allowing a slope discontinuity at the transition between the ramp and the wave data.

As in the previous implementation, after the required output time series has been identified with the augmented input time series, its initial and final parts corresponding to the ramps are removed.

3.2.2   Newton-Raphson Scheme

The algorithm for the Newton-Raphson scheme followed in WAVEMAKER is

|	
1.	Estimate $\mathbf{C}$ from observed history (Eq.s 3.9, 3.13)
2.	Initial Guess $\mathbf{A} = \mathbf{C}$
3.	Estimate $\mathbf{B}$ from $\mathbf{A}$ (Eq.s 3.4, 3.6, 3.13)
4.	Find $\mathbf{f(A)}$ (Eq. 3.14)
5.	Find $[J]$ (Eq.s 3.20, 3.21)
6.	Solve $[J]\mathbf{h} = -\mathbf{f(A)}$ to find $\mathbf{h}$
7.	Update $\mathbf{A} = \mathbf{A} + \mathbf{h}$
8.	Check Convergence (see next section): If converged terminate else go to 3
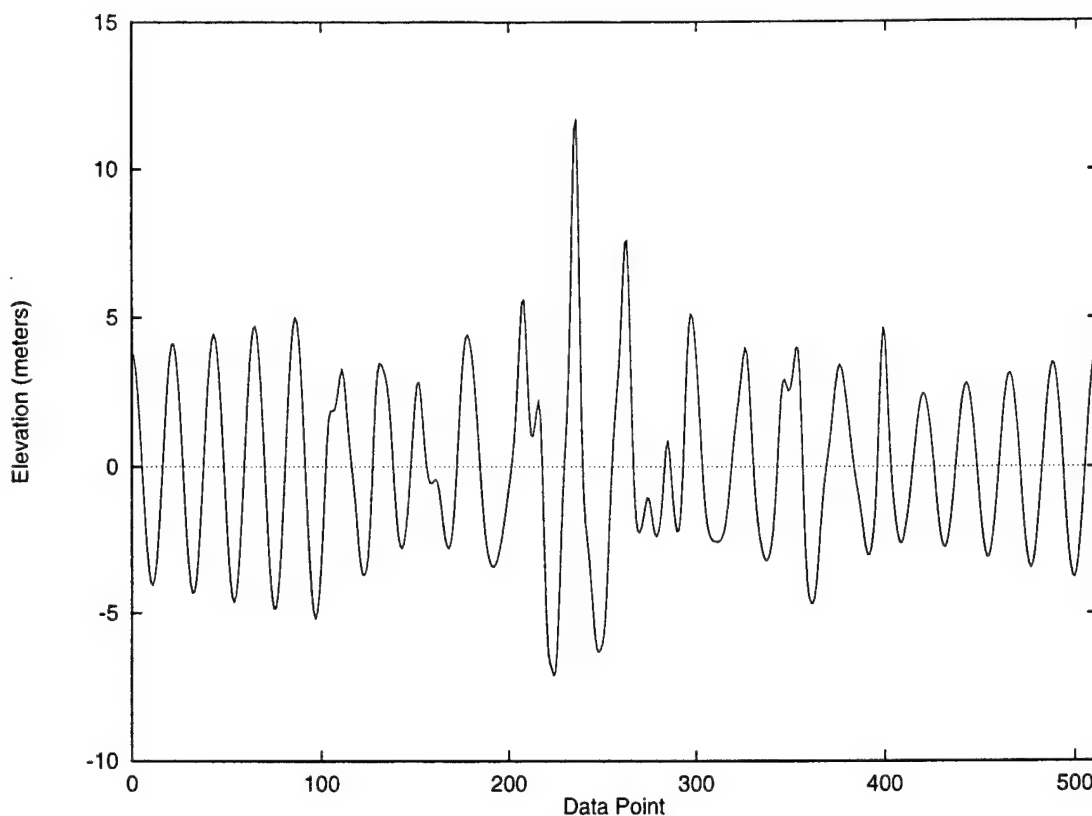
Figure 3.2: Single Data Window including Ramps as generated in Revised WAVE-MAKER 3.2. The first and last 100 data points are artificially generated ramps.

3.2.3 Convergence Criteria

The Newton-Raphson iteration scheme is terminated based on the following conditions:

- **Program Converged**: If the rms of the increment vector $\mathbf{h} = \sqrt{\sum_1^N h_k^2/N}$ is less than a specified tolerance, the program is said to have converged. This convergence tolerance is specified as a fraction α ($= 0.0001$ in WAVEMAKER) of the standard deviation of the observed wave history $\sigma_{\eta,\text{obs}}$.
- **Program Diverging**: If the rms of the identified first-order history $\sigma_{\eta,1}$ at any iteration p is larger than a specified fraction β ($= 200$ in WAVEMAKER) of $\sigma_{\eta,\text{obs}}$ then the identification scheme is restarted with a smaller initial guess which is a truncated and scaled down version of $\mathbf{C}$. The truncation point is at twice the peak spectral frequency of $\eta_{\text{obs}}(t)$ and the scaling factor is $factnu^r$ ($= 0.9^r$ in WAVEMAKER), where r is the number of restarts

needed so far. Thus the restart guess in complex Fourier notation is

$$X_k = \begin{cases} factnu^r Z_k & ; & \omega_k < 2\omega_{\text{peak}} \\ 0 & ; & \text{otherwise} \end{cases} \tag{3.25}$$

- **Maximum Iterations Reached**: If the maximum allowed iterations, specified by the variable *mxiter* (= 10 in WAVEMAKER), is reached and the program has still not converged, then the program restarts the Newton-Raphson identification scheme with a smaller initial guess = *factnu^r* **C**.
- **Maximum Restarts Reached**: If the maximum allowed restarts, specified by the variable *nuiter* (= 5 in WAVEMAKER), is reached then the program terminates the identification scheme in the present window and proceeds to identify in the next observed history window.

3.2.4   Implementation

The first-order components for the observed wave history, of length N_{obs}, are identified in contiguous windows, each of length $N \leq N_{\text{obs}}$. The identification analysis is performed in this way to minimize the computer memory usage by WAVEMAKER. Recall that the Jacobian matrix $[J]$ is a $N \times N$ matrix and the memory usage is directly governed by the matrix size of this variable. In principle, if there is sufficient memory we could set $N = N_{\text{obs}}$ and identify the first-order component for the entire observed history in one window, however, this is not usually not the case and we resort to identifying in contiguous windows, as shown in Fig. 3.3.

The first-order component is identified independently in each of the windows in sequence. The last window is skipped if its contains points less than N.

3.3   Input Specification

The input specification for the identification of first-order wave process is in a command-line format similar to the simulation input. A typical input file for identification is:
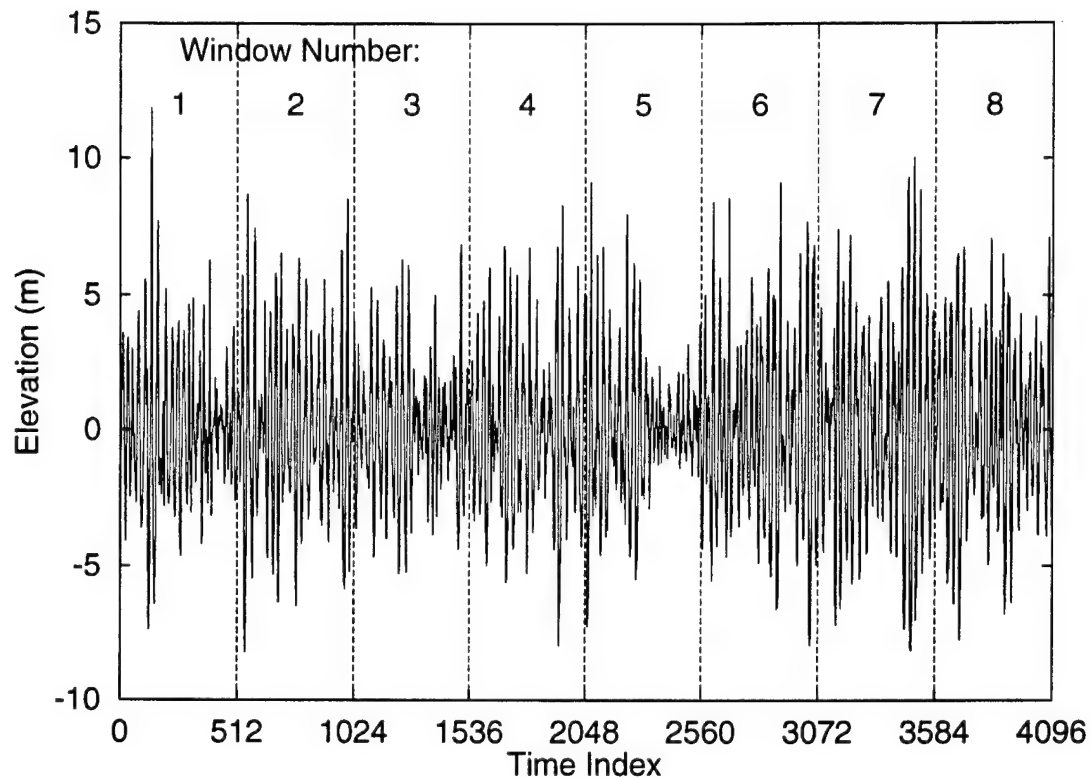
Figure 3.3: Identification of first-order wave components is done in contiguous windows of the observed history

```
# Typical input file: syntax description

identify filename dt winsize nramp
depth value
define varlimit value
define gravity value
write history filename1 filename2
```

Typical input file: syntax description

Any line beginning with a "#" symbol is treated as a comment line and is ignored. Blank lines in the input file are ignored, as well.

define varlimit *value*
define gravity *value*

The keywords **varlimit** and **gravity** have the same meaning as in the simulation section and the user is referred to Section 2.3 to understand the use of these commands.

depth *value*
The keyword **depth** as in the simulation section indicates the water depth at which the identification analysis is to be performed.

identify *filename dt winsize nramp*
The keyword **identify** indicates to the program that the user intends to identify the underlying first-order wave history for a given observed wave history. This command requires three arguments which in sequence are:

- *filename*: The name of the file, a character string, containing the observed wave time history for which the underlying first-order wave history is to be identified. The data in the **first** column in *filename* is read as the observed wave time history. Any blank lines in *filename* or lines that do not begin with a number are ignored.
- *dt*: This value, a real number, indicates the time resolution of the wave history provided in *filename*. In other words, *dt* is the time difference between two successive elevation values in the observed wave history.
- *winsize*: An even integer value indicating the window size or the number of points of the provided wave history to be used in each Newton-Raphson iteration. The first-order wave components are identified in windows (of size *winsize*) in sequence for the provided time history. If the last window contains number of points less than *winsize*, then *winsize* is reset to the remaining number of points. If the original history contains an odd number of points then the last point is ignored.
- *nramp*: An integer value indicating the number of points to be included at each end of the portion of the wave history in each window. The number of data points processed in each window will be *winsize*−2×*nramp*. If *nramp* is not included after the keyword *identify* or is set to zero, then no ramps are included.

The maximum value of *winsize* is *mlen*, which is set to 1024 points in WAVEMAKER. This limitation on *winsize* can be changed according to the user's needs or the computer's limitations by modifying the value of *mlen* in the program source code and and recompiling. This maximum number of points includes two times the number of ramp points. Note that we require *mlen* < 2×*mxgrd* in the program. These dimension values are set in this way so as to minimize the memory requirements of WAVEMAKER.

write history *filename1 filename2*
The command **write history** is used to specify the file names where the identified histories are to be written. The identified first-order wave history is written in file *filename1* and the identified second-order, combined first- and second-order, and the observed wave histories are written in file *filename2*. Default values assigned to *filename1* and *filename2* are **gauss.ide** (**Gauss**ian **ide**ntification) and **ngauss.ide**, respectively.

3.4   Output Format

The output file names are governed by the command

write history *filename1 filename2*

with default names being **gauss.ide** and **ngauss.ide** for *filename1* and *filename2*, respectively.

filename1 contains the identified first-order wave history in a column of real numbers. Each line of this file contains one real number indicating the elevation of the first-order wave history (see example output files in Appendix C). The time resolution of this first-order history is dt, equal to the dt provided in the input file using the command **identify**. This file also contains comment lines that begin with a "#" symbol as the first character on the line. The first comment line contains information on the contents of the file, and the following comment lines contain 3 integers: the first is the window number being identified, the second is the number of iterations required for convergence, and the third is the number of restarts needed for convergence.

filename2 contains the second-order correction, the combined first- and second-order waves, and the observed wave time history. The second-order correction is found from the identified first-order waves, and these two are added together to yield the combined second-order wave history. These histories are provided in three columns in *filename2*, or in other words each line contains three real numbers: the first is the second-order wave elevation, the second is the combined identified wave elevation, and the third is the observed wave history (see example output file in Appendix C). A match of the total identified and the observed wave histories will verify successfully identification by WAVEMAKER. The time resolution of each of these histories is dt. This file also contains comment lines beginning with a "#" symbol that provides information similar to the comment lines in *filename1*.

3.5   Examples

In this section we present two sample problems to illustrate the use of the identification capabilities of WAVEMAKER. Example 1 is based on the example presented in the simulation chapter. Sample input and output files of this identification example are included in the distribution diskette. Example 2 presented here demonstrates the identification of first-order components of a measured wave tank history. Note that sample input or output files of this second example are not included in the distribution.

3.5.1   Example 1

The example of the simulation capabilities of WAVEMAKER involved simulating a second-order wave history characterized by a JONSWAP spectrum with $H_s = 12$ [m], $T_p = 14s$ and $\gamma = 3.3$ in 70 [m] water depth. We will use the combined second-order simulated history and try to identify its first-order wave component and compare it to the input first-order component used to simulate the combined wave history. The input file for the identification run is

```
# Wave Identification Input File

identify hist.dat 0.5 512 100
depth 70.0
write history gauss.ide ngauss.ide
define varlimit 0.01
define gravity 9.807
```

Alternatively, if we intend to use the default definitions in the program then our input file could be (this will produce the same output as the extended version of the input file):

```
# Wave Identification Input File
# (Alternative format)

identify hist.dat 0.5 512 100
depth 70.0
```

The input file **hist.dat** contains a column of real numbers (see sample files listed in the appendix) which will be read in as the observed wave history. In this example, the input file **hist.dat** contains the second column of the results produced in the Simulation example output file **ngauss.sim**. The first-order components will be identified for this observed history and placed in the file **gauss.ide**. The corresponding second-order components, the combined identified history and the observed wave history are written in the file **ngauss.ide**.

Output File	Contents
gauss.ide	Identified First-Order Time History
ngauss.ide	Corresponding Second-Order Time History

Figure 3.4 shows the observed wave spectrum and the identified first-order spectrum along with the corresponding second-order wave spectrum. We see that small second-order contribution to the power spectrum, roughly a decade below the first-order spectrum even at frequencies twice the peak spectral frequency, suggests the difficulty in identifying these components. Figure 3.5 shows the observed wave history and the identified first-order wave history in cycles around the maximum crest height. Compare this to the simulation example where we solve the forward problem of finding the combined (first- plus second-order) history from a given underlying first-order wave spectrum. The identified first-order component in Fig. 3.5 is almost the same as the underlying first-order component (denoted Gaussian) in Fig. 2.2 and these two are shown together in Fig. 3.6. Note how close to each other the two first-order components are; any numerical differences can probably be further reduced by using a larger window size (greater than 512, for example) in the identification scheme.

3.5.2 Example 2

In this example we will identify the underlying first-order wave component for a measured wave tank history that reflects a water depth of 175m. For this example the measured history is located in file **wave.dat** and has a $dt = 0.3354$ seconds. We will use windows of *winsize* = 512 to identify the first-order components. The input to WAVEMAKER is:

```
# Wave Identification Input File
# using default definitions

identify wave.dat 0.3354 512 100
depth 175.0
```
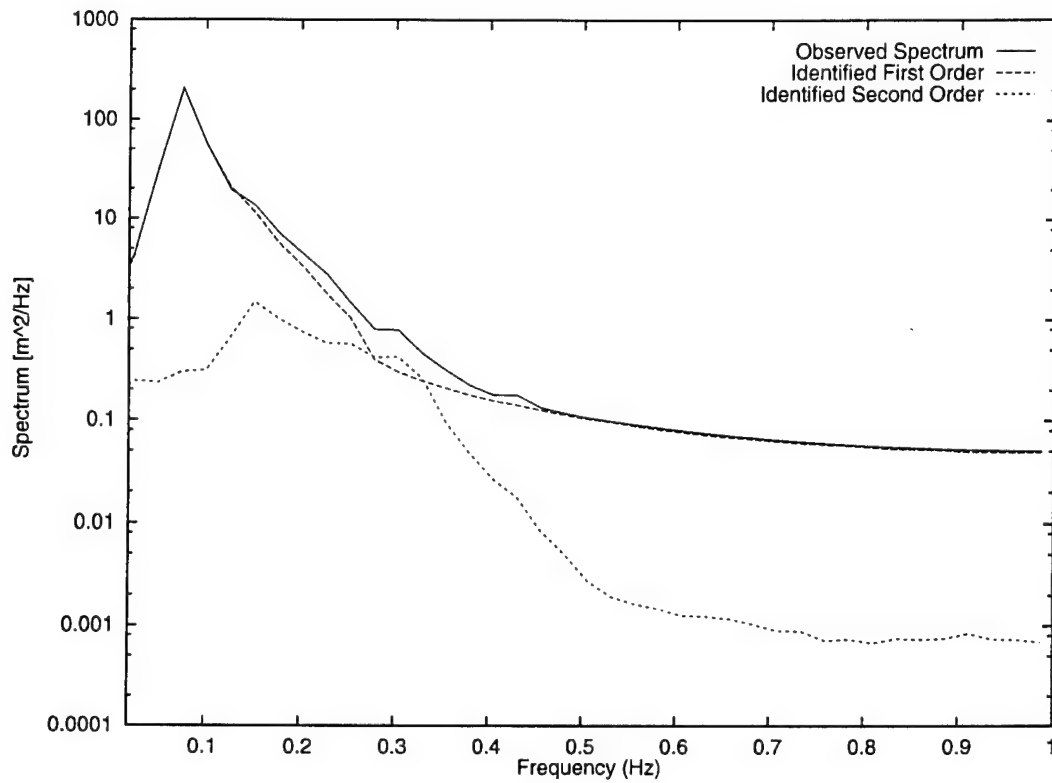
Figure 3.4: Wave spectrum: observed vs. identified first- and second-order (Example 1)
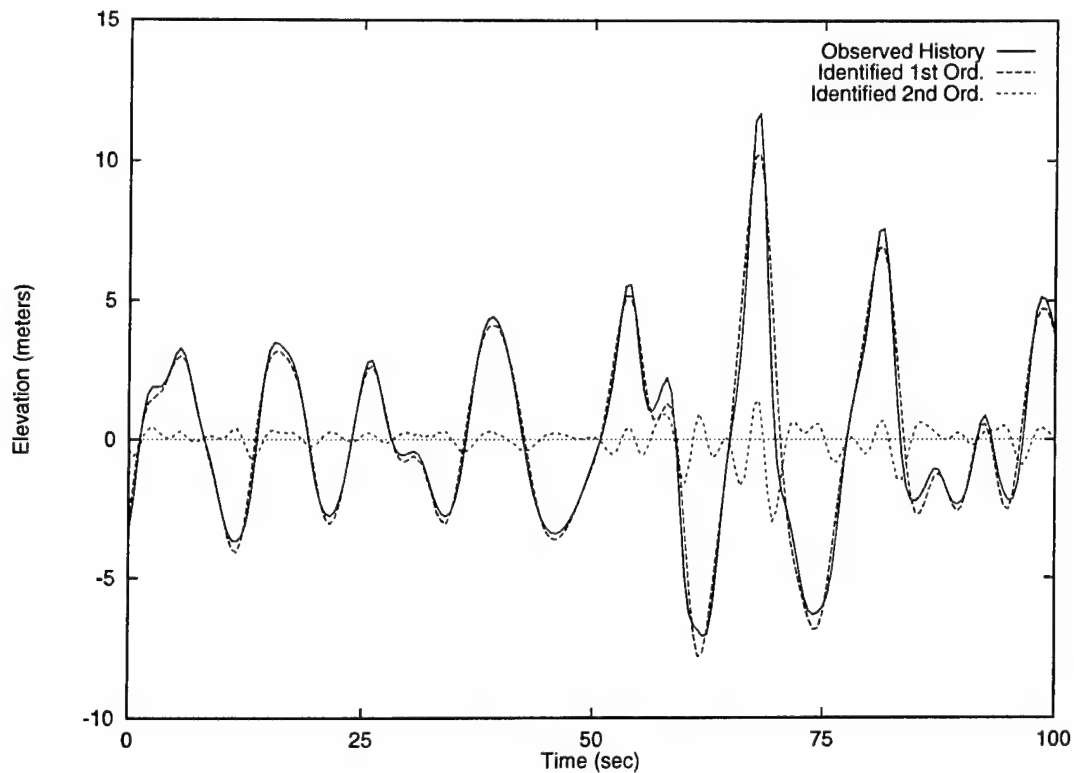
Figure 3.5: Wave history: observed vs. identified first- and second-order (Example 1)
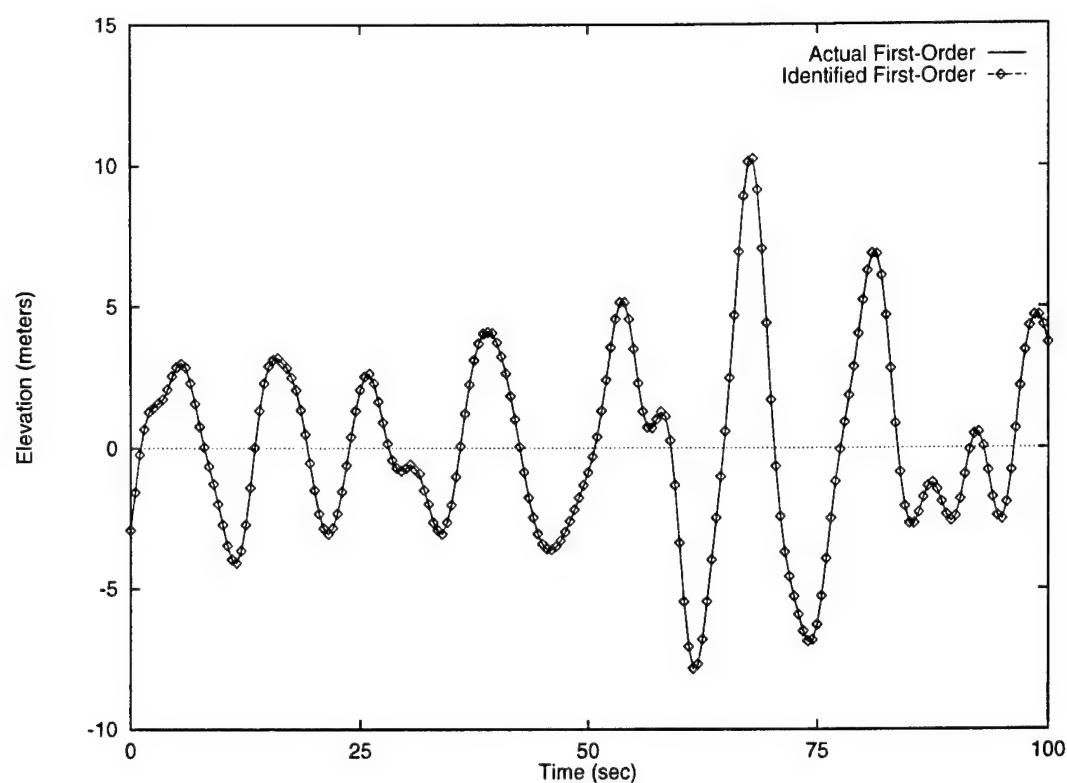
Figure 3.6: Identified first-order vs. actual first-order wave history (Example 1)

Figure 3.7 shows a portion where the maximum crest height occurs in the measured wave tank history. The figure also shows the identified first-order and the corresponding second-order wave histories. Note how the second-order wave component affects the first-order peaks, amplifying the crests and moderating the troughs. Figure 3.8 shows the wave spectra for the measured history along with the first-order and the second-order spectra. Again, observe that the second-order energy is significantly small compared to the first-order, however, phase locking of the first- and the second- component (Fig. 3.7) leads to larger crests and flatter troughs.
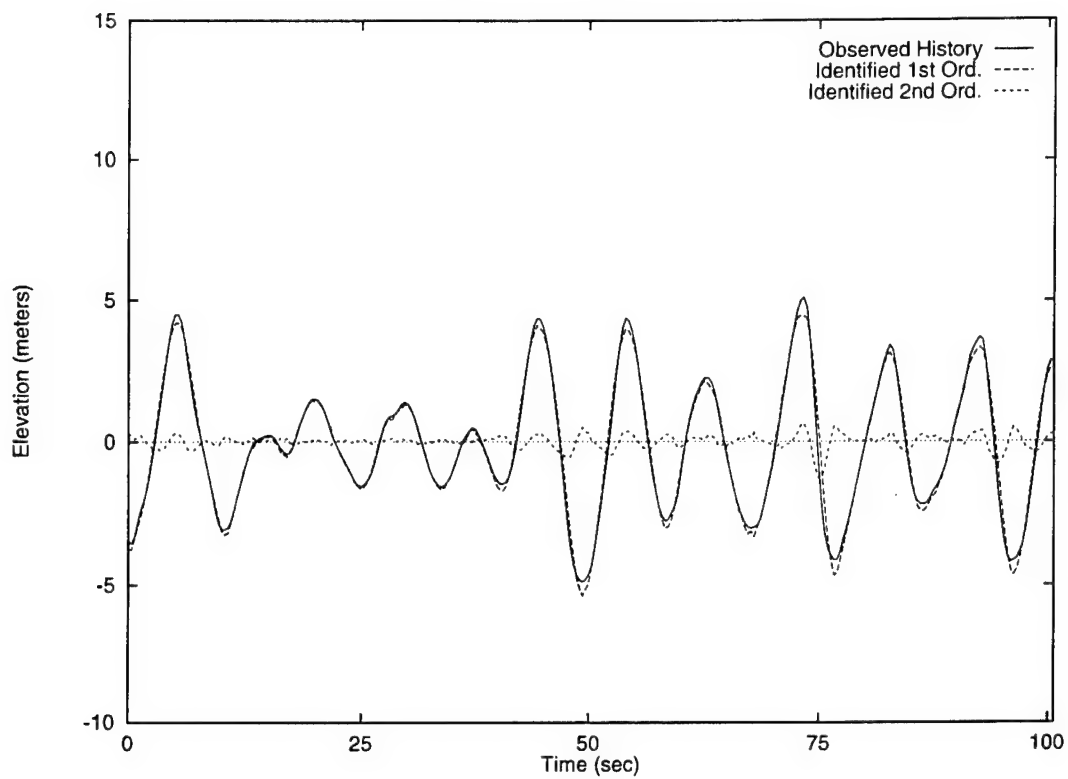
Figure 3.7: Wave history in wave tank: observed vs. identified first- and second-order (Example 2)
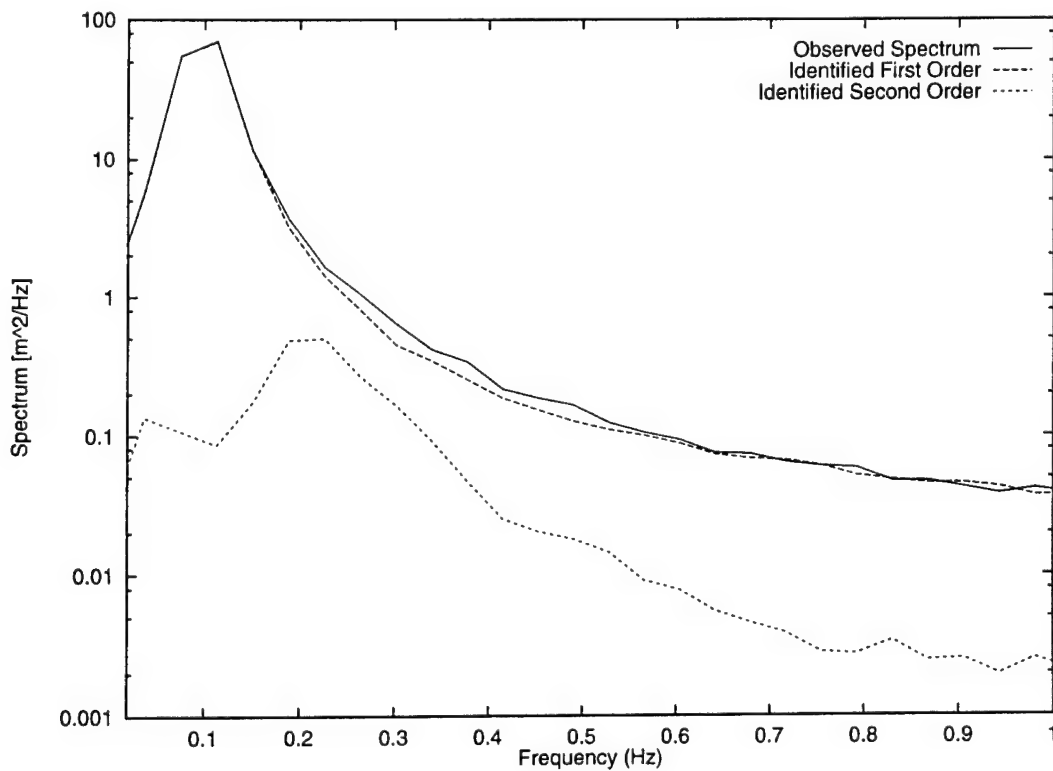
Figure 3.8: Wave spectrum in wave tank: observed vs. identified first- and second-order (Example 2)

Chapter 4

Prediction of Second-Order Random Waves

4.1   Introduction

Chapters 2 and 3 of this document outline the *Simulate* and *Identify* options of WAVEMAKER. The *Predict* option builds on the same theory and implementation to provide another capability: to propagate an observed input time series through a second-order model to produce the first- and second-order outputs at multiple fixed spatial locations.

Chapter 2 describes a procedure for simulating a second-order random wave process corresponding to a user-specified wave power spectral density. These wave histories include a second order Stokes correction at the sum and difference of each wave frequency pair, as calculated in the frequency domain. An inverse FFT is applied to this frequency domain representation of the wave process to produce a time-history. Time histories for multiple spatial locations are generated by applying the linear dispersion relation in the frequency domain of the underlying first order wave process prior to inclusion of the second-order correction at each spatial location.

Chapter 3 describes a procedure for identifying an underlying first-order process from a user specified wave history. This identification, an inverse feature to simulation, is based on a Newton-Raphson scheme to solve N simultaneous nonlinear equations to identify the first-order waves which, when run through the second-order wave predictor, matches the observed waves.

This chapter describes a third execution option for the WAVEMAKER program. The

Predict option uses the underlying first-order wave history generated by the identify option at one location as input to generate consistent first and second order wave time histories at other locations using essentially the same theory as the simulate option. Thus, the Predict option combines the utility of the procedures developed in Chapters 2 and 3.

4.2   Overview

Measured wave input data, such as that collected with a wave probe, is first identified as described in Chapter 3. The Identified underlying first-order wave process is used as the input. An FFT procedure generates a discrete Fourier sum representation of the wave history in the frequency domain. This frequency domain representation of the wave process is equivalent to that generated in Chapter 2, except that the sum now corresponds to a user-specified wave time history, rather than a user-specified power spectral density. Time histories for multiple spatial locations are generated by use of the dispersion relation with a second order correction as described in Chapter 2.

The following flowchart shows the flow of the calculation for the predict option. The user specifies an input wave history, which will generally be a measured wave. Identify, as described in Chapter 3, is used to separate the total process into consistent first and second order processes. The linear dispersion relation is applied to the consistent first order part of the wave history, which produces an equivalent first order history at user specified locations. The predict option then applies the inverse of the Identify process such that the consistent second order part of the wave history is recreated at the new locations. It is the use of the same "black box" in the separation (identify) and recreation (predict) calculations that ensures the newly created second order part of the wave histories at the new locations are consistent with those identified at the original input wave location.

The implementation of the prediction capability differs from simulation only in the first step. In simulation, complex Fourier amplitudes of the input were generated from the PSD model. In prediction, they are generated by simply FFT'ing the observed input time series. Once the input is described by complex Fourier amplitudes, the procedure is identical to that of simulation.

This tool provides an easy way to compare model test results with real-world observations or to compare specific model test results such as airgap measurements at a known location with a model test reference wave history.
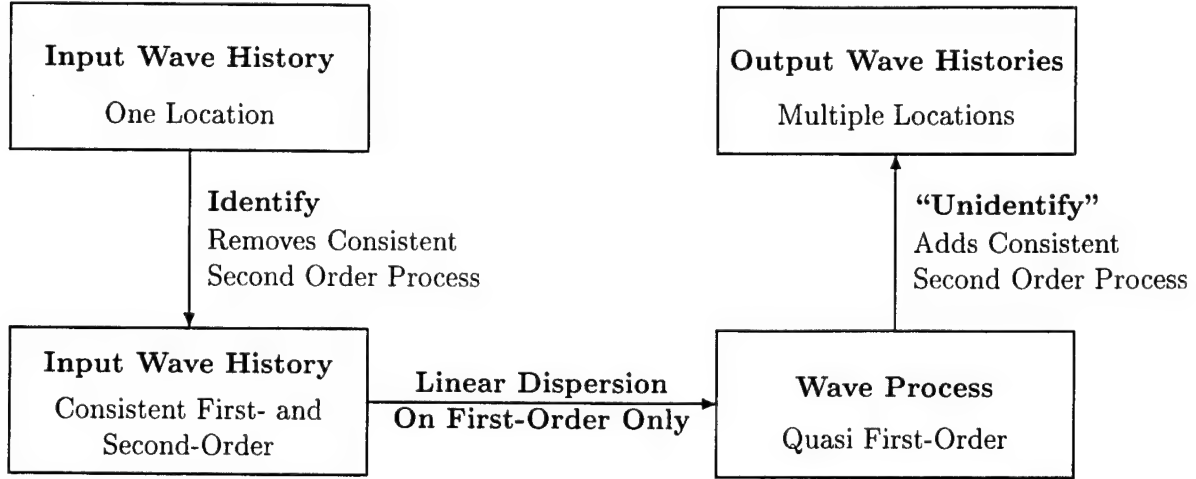
Figure 4.1: Overview of Prediction Calculation Flow

4.3 Methodology

4.3.1 Underlying Theory and Assumptions

We first consider $\eta_1(t)$, the first-order wave elevation, at a specific reference location (say $x{=}0$). For either frequency-domain analysis or time-domain simulation, it is convenient to write $\eta_1(t)$ as a discrete Fourier sum over positive frequencies ω_k:

$$\eta_1(t) = \sum_{k=1}^{N}(npts/2)A_k\cos(\omega_k t + \theta_k) = \text{Re}\sum_{k=1}^{N}A_k e^{i(\omega_k t + \theta_k)} \quad (4.1)$$

For purposes of prediction, the lowest frequency interval $d\omega$ is governed by the total period T of the simulation, while the highest frequency is governed by the time interval between data points in the input wave history:

$$d\omega = \frac{2\pi}{T} \quad (4.2)$$

The second-order wave at this elevation, $\eta_2(t)$, is calculated from $\eta_1(t)$ as

$$\eta_2(t) = \eta_1(t) + \Delta\eta_2(t) \tag{4.3}$$

in which $\Delta\eta_2(t)$ includes second-order corrections at sums and differences of all wave frequencies:

$$\Delta\eta_2(t) = \text{Re} \sum_{m=1}^{N} \sum_{n=1}^{N} A_m A_n [H_{mn}^{-} e^{i[(\omega_m - \omega_n)t + (\theta_m - \theta_n)]} + H_{mn}^{+} e^{i[(\omega_m + \omega_n)t + (\theta_m + \theta_n)]}] \tag{4.4}$$

4.3.2   Implementation

The prediction method requires an input time history representing the water surface elevation measurements. The measurements are required to be at evenly spaced time intervals to take advantage of discrete FFT techniques.

An FFT is applied to the time history to calculate consistent values of A_k and θ_k in Eq. 2.1.

Eq. 2.1 is then rewritten as

$$\eta_1(t) = \sum_{k=1}^{npts/2} A_k \cos(\omega_k t + \theta_k) = \text{Re} \sum_{k=1}^{npts} X_k e^{i\omega_k t} \tag{4.5}$$

Here the X_k are complex Fourier coefficients. The lower half of these directly reflect values of amplitude A_k and phase θ_k consistent with the user specified wave history at frequency $\omega_k = k \cdot d\omega$:

$$X_k = \frac{1}{2} A_k e^{i\theta_k} ; \quad k = 1...npts/2 \tag{4.6}$$

The upper half are the complex conjugates of the lower half due to the double sided nature of the FFT in equation (eta1) (the symbol "*") of the lower half:

$$X_{npts-k} = X_k^* ; \quad k = 1...npts/2 \tag{4.7}$$

This reflects that unique information is contained only the lower-half frequencies; indeed, any information in the upper half frequencies (above the Nyquist) is obscured by aliasing.

4.4 End-Point Continuity of a Wave Record

4.4.1 Predict Option

Actual measured waves are generally not continuous in slope or offset between the ends of the measured wave record. Subsequently, the identified first order wave process used as the input to the Predict option is also discontinuous between its ends. This discontinuity could introduce difficulty in the application of FFT techniques because of the "wrap-around" effect inherent to the FFT process.

If an end discontinuity is present, the FFT attempts to duplicate the time-domain discontinuity by introduction of excessively large high frequency components to the frequency domain characterization of the wave profile. These high frequency components are introduced because of the end discontinuity, but are propagated through the entire wave record. After application of the Inverse FFT, the resultant time trace has the spurious high frequency content spread throughout the wave history. There is no damping in the dispersion relation so any numerical noise introduced by end discontinuities is not dissipated as it is propagated to other spatial locations as part of the wave process.

To avoid the introduction of an end point discontinuity, a portion of a wave cycle is artificially generated by the program in this Predict option. First, the period of a characteristic wave cycle, T_{char}, is calculated by averaging the wave upcrossing periods for the first and last ten cycles in the wave record. How large a portion of a single characteristic wave cycle is required for continuity is determined by estimating the "phases" of the waves at the beginning and ending of the wave record as Θ_1 and Θ_2:

$$\Theta_1 = \arctan\left(\frac{\dot{x}_1/\omega}{x_1}\right)$$

and

$$\Theta_2 = \arctan\left(\frac{\dot{x}_2/\omega}{x_2}\right)$$

where

x_2 = Average offset of the last two points in the wave record from the mean

$\dot{x}_1$ = Slope between the first two point in the wave record

$\dot{x}_2$ = Slope between the last two points in the wave record

The portion of the characteristic wave cycle to be included is then the absolute
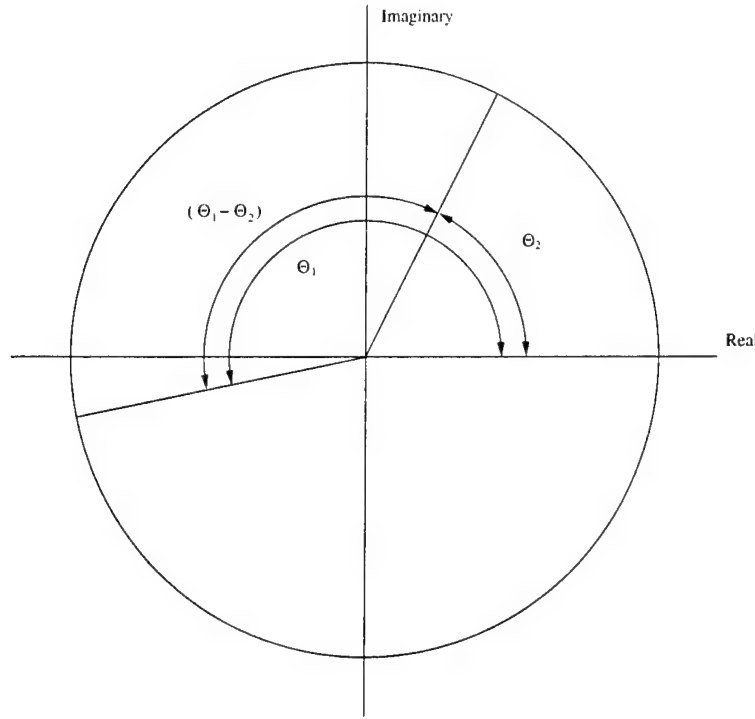
Figure 4.2: Phase Calculation for Added Portion of Wave Cycle

value of $(\Theta_1 - \Theta_2)$ and so the required additional time is given by:

$$T_{added} = \frac{1}{\omega} \left[\arctan\left(\frac{\dot{x}_1/\omega}{x_1}\right) - \arctan\left(\frac{\dot{x}_2/\omega}{x_2}\right) \right] \qquad (4.8)$$

where $\omega = 2\pi/T_{char}$

The required number of additional time points is then $T_{added}/$ (time between successive time points)

A third-order polynomial is then fit through the last two points in the wave record such that the newly created polynomial would be continuous with the first two points on the record if the wave record repeated. Thus, a small number of artificial data points (necessarily less than one wave cycle) is generated which makes the wave record continuous between its end-points so no spurious high frequency energy is introduced into the FFT and IFFT calculations.

Note that when predictions are made for locations other than the location of the

original wave probe, the artificially generated wave data is shifted according to its Fourier frequency content, as is the rest of the data. Thus, part of the artificially generated data is shifted a short distance into the body of the actual measured data, from which it generally can not subsequently be removed.

4.4.2 Simulate Option

End discontinuities are not problematic in the Simulate option because the Simulate option generates waves by use of Fast Fourier Transform techniques. The generated wave components have the convenient property that the resultant time trace of each of the component waves is continuous in slope and offset between the two ends of the generated time series. This property enables the generation consistent simulated waves at multiple spatial locations without requiring special treatment of end discontinuities.

4.4.3 Identify Option

End discontinuities could be problematic in the *Identify* option. The identification process is extremely computationally intensive. The input wave time history is separated into smaller time history segments; each of these windows generally has its length set to an even power of two to reduce the total computational intensity of the FFT process. Discontinuities will generally be present between the ends of these windows.

A different approach is applied to gain end-point continuity in the *Identify* option such that the length of each window can be maintained as an even power of two. The approach is to increase the length of each window by introduction of artificially generated points on each end of the each wave history segment. The number of artificial data points plus the number of actual wave data points are selected such that the total remains an even power of two.

The artificial data is generated such that the average wave upcrossing frequency of the actual wave segment is approximated by the artificially generated waves. The amplitude of the artificial waves is gradually reduced (or increased) to the mean amplitude of the peaks of the data in the current window. The window has the appearance of a segment of actual wave data surrounded at each end by a "ramp" of artificial data. Continuity between the ends of the window is assured because each end has "waves" of the same phase and amplitude and nearly the same frequency.

The dispersion relation is not applied in the Identify option, so dispersion of the ramps into the wave record is not a concern. The artificial data is removed from the ends of each segment prior to concatenation of the segments into the identified first and second order wave histories. The methodology of the ramp implementation was previously discussed in Section 3.2.1.

4.5   Input Specification

The input specification for the prediction of a first-order wave process is in a command-line format similar to the input for the simulation or identification options.

Typical input file: syntax description

```
# Wave Prediction Input File

predict filename npts dt
define gravity value
define varlimit value
depth value
write history filename1 filename2
location nloc
value1
value2
   ⋮
valuenloc
```

Typical input file: syntax description
Any line beginning with a "#" is treated as a comment line in the input file and is ignored by the program. Blank lines are also ignored by the program. As with other options of Wavemaker, most lines in the input file can be entered in any sequence. There are two exceptions: the first non-comment line must indicate the type of analysis to be performed (simulate, identify or predict), and the second is that the location specification must be the last set of lines in the file.

predict *filename npts dt*
The required keyword **predict** indicates the user intends to predict the wave history at another spatial location based on a (measured) wave history. This command requires three sequential arguments:

- *filename:* A character string indicating the name of the file containing the observed or identified wave time history for which wave predictions at other locations are to be performed. The data in the first column in filename is the wave time history. Any blank lines in filename or lines that do not begin with a number are ignored.
- *npts:* An integer number indicating the number of data lines to be read from the input wave file. npts times dt equals the elapsed time. If npts is specified to be a number smaller than the number of lines in the data file, the first npts data points in the file are used..
- *dt:* A real number indicating the time resolution of the wave history provided in filename. dt is the difference between measurement times of two successive data points in the input wave history.

define
The optional keyword **define** is to be followed by another keyword indicating which constant is to be defined.

define varlimit *value*
define gravity *value*
The keywords **varlimit,** and **gravity,** have the same meaning as in the simulation and identification sections.

depth *value*
The required keyword **depth,** as in the simulation option, indicates the water depth at which the identification analysis is to be performed. *value* is a real number. As in the simulate and identify options, **depth** can be specified in any units consistent with the units of gravity specified using the define keyword. If the user does not explicitly specify a value for gravity, the default value 9.81 meters/sec^2 and then the depth value should be specified in meters.

write history *filename1 filename2*
write statistics *filename3 filename4*
The optional keyword **write** is to be followed by either the keyword **history** or the keyword **statistics. write** is used to specify the files to which the predicted time histories or summary statistics are to be written.

write history *filename1 filename2*
The final length of each of these files should be the same as the length of the original input wave history, plus a small amount of descriptive text. Output file formats are covered in Section 4.6 and in more detail in 2.4.1.

filename1 is the file to which the predicted first-order wave histories for each of the

specified locations are written.

filename2 is the file to which the predicted second-order wave histories for each of the specified locations are written.

Default values are assigned to *filename1* and *filename2* if none are specified by the user. These default filenames are **gauss.pre** (**Gauss**ian **pre**diction) and **ngauss.pre**, respectively.

write statistics *filename3 filename4*
The final length of each of these files should be the same as the number of output locations specified in nloc below, plus a small amount of descriptive text. Output file formats are covered in section 4.6 and in more detail in 2.4.1.

filename3 is the file to which summary statistics of the predicted first-order wave histories for each of the specified locations is written.

filename4 is the file to which summary statistics of the predicted second-order wave for each of the specified locations are written.

Default values are assigned to *filename3* and *filename4* if none are specified by the user. These default filenames are **gauss.pst** (**Gauss**ian **p**rediction **st**atistics) and **ngauss.pst**, respectively.

location *nloc:*
The required keyword location is used to specify the number of spatial locations at which both the first- and total second-order wave histories are to be predicted.

nloc is an integer specifying the desired number of locations. The maximum allowable number of locations is 50.

value1
value2
  ⋮
valuenloc

Each of these real numbers indicates one spatial location measured in the along-wave direction at which prediction results are to be generated. Both negative and positive values are acceptable. Positive values are down-stream of the original input wave history; negative values are closer to the wave generating device. As noted earlier, this specification of the spatial locations must be after all other input in the

file. Also, the last location, *nloc*, must be followed by a carriage return; i.e. the input file must end on a new line.

It is suggested that one of the output locations be 0.0, where the waves were originally measured. It can then be confirmed that the wave data has been correctly read and interpreted by the program by verifying that the predicted wave history at the wave probe equals the target wave history measured at the same location.

4.6 Output Format

A total of four output files are produced by the driver program. Two output files contain the time histories: one for the underlying first-order wave histories and the other for the total second-order histories. The other two output files contain wave statistics: first four moments, minimum and maximum. Again, results for the first- and second-order wave histories are separated into two files.

4.6.1 Time History Output

As noted previously, by default the first- and second-order histories are written to the files **gauss.pre** and **ngauss.pre**. Other choices of output filenames can be specified by the optional **write history** command. The format of this output depends on the number of spatial locations specified. If the number of locations (*nloc*) is less than or equal to 8 then the output is in **Format1** otherwise the output is in **Format2**. Both of these formats write out 3 header lines beginning with a "#" sign. These are to be treated as comment lines in the output file.

Format1 outputs data in *nloc*+1 columns. The length of each column is equal to the number of points desired in each prediction. The first column contains the time increments in seconds going from 0 to T with dt $= T/npts$. Columns 2 through *nloc*+1 contain the predicted time history values at the specified locations x1, x2, x3, ...xnloc, respectively. Thus, column 2 contains the wave elevation at location x1, column 3 contains wave elevation at location x2, and so on.

Format2 is for handling *nloc* greater than 8. The output begins with the time increment Ti in seconds on a line by itself. The time history values for the specified spatial locations at time Ti are written in the next line onwards, in sets of 10. So if 9 locations were specified (i.e., *nloc* = 9) then the time increment is printed on a line by itself followed by a line containing 9 time history values at that time increment. The

next line contains the next time increment followed by another set of 9 values, and so on. If, on the other hand say 28 locations were specified, then a time increment is written on a line followed by 28 time history values (corresponding to 28 locations at that time increment) in the next 3 lines. The first line of the 3 lines contains 10 time history values for the first 10 locations specified. The next line contains 10 history values for locations 11 through 20 and the following line which is the third line of the set will contain only 8 history values for location 21 through 28.

4.6.2   Wave Statistics Output

The statistics of the simulated histories are also estimated by the driver program. These statistics include the mean, standard deviation, skewness, kurtosis, minimum, and maximum. As noted in the previous section, first- and second-order simulation results are written by default to the files **gauss.pst** and **ngauss.pst**, respectively. The optional command **write statistics** can alter this choice of output filenames.

The output format in both of these files begins with 2 header lines, each of which begins with a "#" sign. The output is in seven columns. The first column specifies the spatial location. The following six columns contain statistics of the wave history at the spatial location specified in column 1. Columns 2 through 7 contain, the mean, standard deviation, skewness, kurtosis, minimum, , and maximum value in that order.

4.7   Examples

In this section we present two sample problems to illustrate the use of the prediction capabilities of wavemaker. Example 1 is based on the examples presented in the simulation and identification chapters. Sample input and output files of this prediction example are included on the distribution diskette. Example 2 presented here demonstrates the prediction of first- and second-order components of a measured wave tank history. Note that sample input or output files of this second example are not included in the distribution.

4.7.1   Example 1

The example of the simulation and identification capabilities of wavemaker involved simulating a second- order wave history characterized by a JONSWAP spectrum with

$H_s = 12$ [m], $T_p = 14$s and $\gamma = 3.3$ in a 70 [m] water depth. We will use the first-order wave history and try to predict the first- and second-order wave history at another location for comparison with the histories directly simulated at the same alternate location (60 meters down-stream). Note that the data points near the ends of the files differ slightly because the end-point discontinuities are handled differently in the predict option than in the identify option, as discussed in Sections 3.2.1 and 4.4. The data-points internal to the file, however, are relatively unaffected by the differing treatment of end-point discontinuities, as shown in Figure 4.3.

The input file for the prediction run is:

```
# Wave Prediction Input File

predict gauss.ide 4096 0.5
depth 70.0
write history gauss.pre ngauss.pre
location 3
0.0
2.0
60.0
```

The input file **gauss.ide** contains a column of 4096 real numbers (see sample files listed in the appendix) having an incremental time step of 0.5 seconds. This file is read in as the first order components of the identified observed wave history.

The linear (Gaussian) prediction are written to the file **gauss.pre**, while the prediction including the non-linear (non-Gaussian) terms is written to the file **ngauss.pre**.

4.7.2 Example 2

In this example we will continue Example 2 from the Identify section by predicting the wave profile at two locations in addition to the wave measurement location. The underlying first-order wave component for a measured wave tank history. For this example the identified history is located in file **gauss.ide** and has a $dt = 0.3354$ seconds.

The input to wavemaker is:
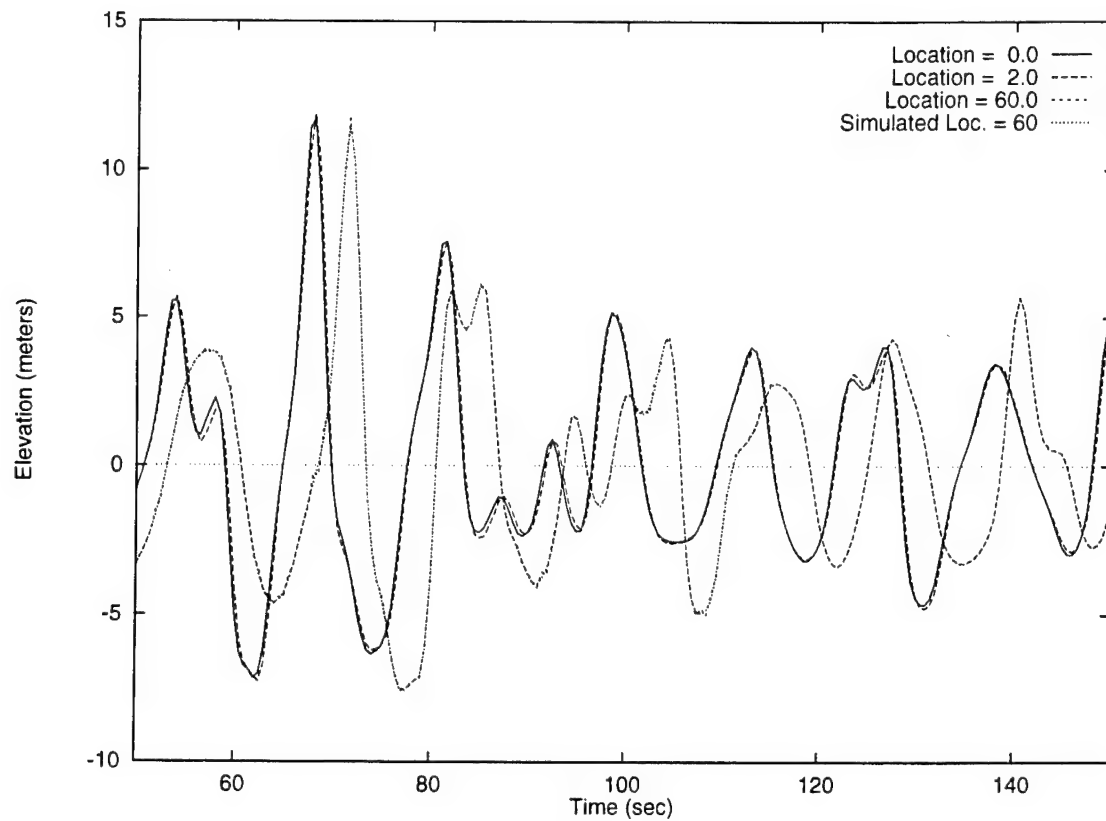
Figure 4.3: Non-Linear Prediction and Simulation Results

```
# Wave prediction Input File

predict gauss.ide 4096 0.3354
depth 175.0
write history gauss2.pre ngauss2.pre
location 3
0.0
2.0
60.0
```

Figure 4.4 shows a portion of the wave history as predicted at three spatial lo-
cations. Note that the shape of the wave profile has changed only slightly when it
is shifted a short distance (2 meters), but has more significant shape changes at the
larger shift (60 meters).
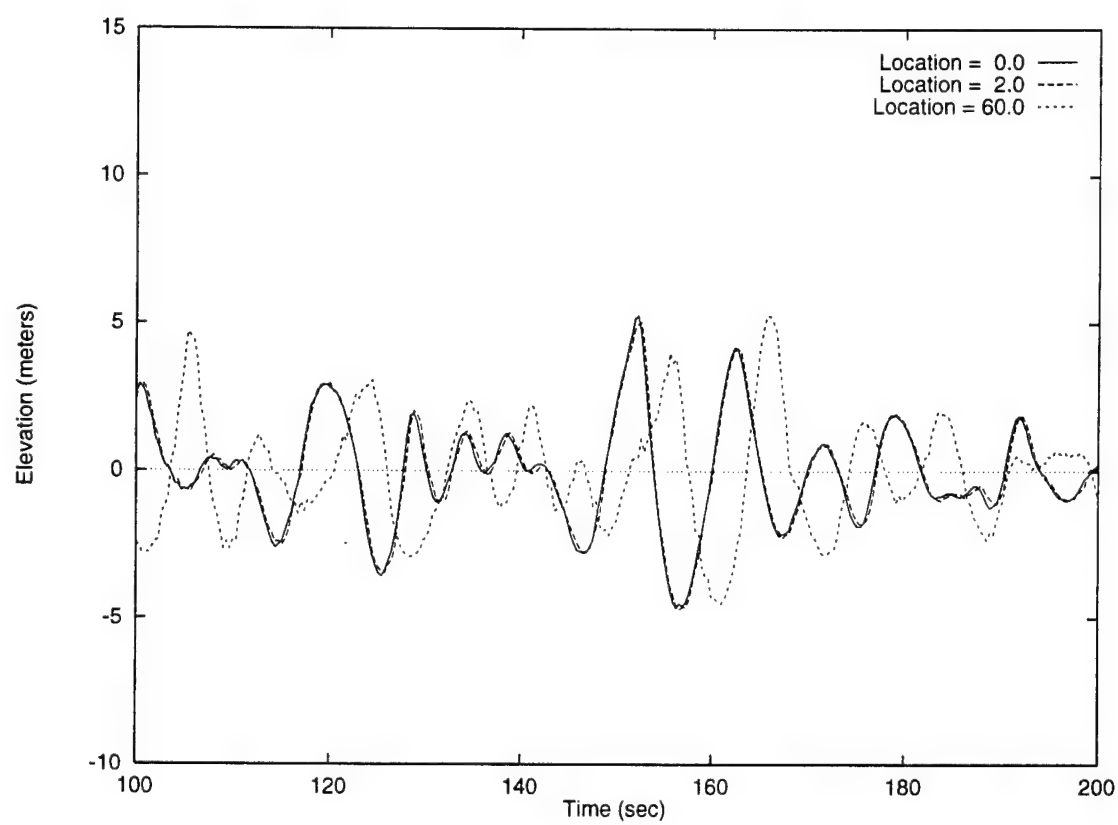
Figure 4.4: Second-Order Predicted Results - Wave Tank Example

Chapter 5

Wave Prediction from a Moving Reference Point

Consider a physical model test in a wave basin during which a local phenomenon such as wave impact on the deck of a semi-submersible or wave amplification near a column is observed. The analyst may wish to assess the local interaction between the wave and the moving structure.

The assessment of this interaction requires knowledge of what the wave would have been had it not been affected by the presence of the structure. Optimally, this undisturbed wave history should be described at specific locations relative to the moving structure: the undisturbed equivalent of measurements taken by wave probes fixed to the moving structure.

The time history developed using the new *translate* option is a prediction of ocean waves as they would be observed from a moving vessel if the vessel were forced to move horizontally as described by the user-specified motion time history and if there was no interaction between the vessel and the waves.

5.1 Methodology

Prediction of wave histories from a moving reference point builds on the previous capability of WAVEMAKER, which predicts wave histories at fixed reference points based on both first- and second-order random wave theory. To develop a wave time-history with a moving reference point, wave time-histories are developed at a series of evenly spaced fixed locations between the maximum and minimum excursion of the vessel.

The wave history at a location fixed to the moving vessel is found by spatial interpolation at each point in time between the calculated wave histories at the regularly spaced fixed locations.

The user must specify a wave history at a known location from which the series of wave histories will be generated, and a motion time history indicating the horizontal motion of the vessel in the direction of the waves. As in other cases of the *predict* option, WAVEMAKER takes this input wave history to be the underlying first-order Gaussian part of the wave; it then constructs a second-order correction due to all sum and difference frequency contributions in the irregular input wave. The necessary input wave history can be either an observed history from which the Gaussian part has been identified using the *identify* option of WAVEMAKER, or the Gaussian part of a wave generated using the *simulate* option of WAVEMAKER. The user may also specify the number of regularly spaced wave time histories to be generated for later interpolation, and may specify use of either linear or cubic interpolation.

5.2   Input Format

Use of the *translate* option is specified by the **translate** command line which can be anywhere in the input file after the **predict** command line other than the last line of the file which is reserved for the location specification. The syntax of the command line is:

translate *filename ntrack interp*

where:

translate indicates the option to be exercised.

filename indicates the name of the file containing the user-specified motion time history of the vessel. The file is to contain one column of real numbers, each of which indicates the instantaneous location of the vessel relative to the fixed point in space at which the wave history was specified in the *predict* command. Positive is in the direction of wave travel.

Various reference points fixed to the vessel can be specified at the end of the input file after the *location* command.

ntrack is an optional integer number indicating how many regularly spaced wave histories are to be generated for purposes of interpolation. This input should be

specified with some care: the execution time of the program increases linearly with this number, but too small a number will result in meaningless results. The default value of *ntrack* is 10, which is appropriate when horizontal motions are relatively minor.

interp is a character input with allowable values of **"linear"** or **"cubic"** indicating the type of interpolation to be performed between the *ntrack* locations. Linear interpolation will generally yield more stable results and is recommended.

The default output filenames are **gauss.trn** (**Gauss**ian **tr**a**n**slation) and **ngauss.trn** for the resultant time histories and **gauss.tst** (**Gauss**ian **t**ranslation **st**atistics) and **ngauss.tst** for the statistics of these histories. Filenames can be modified as described in Section 4.5. The formats of the output files are as described in Sections 2.4 and 4.6.

5.3 Examples

This chapter offers two examples: Example 1 is a continuation of the ongoing example presented in Chapters 2 through 4. Example 2 is a natural demonstration of the program using artificially generated wave and vessel motion histories.

5.3.1 Example 1

As noted earlier, the *translate* option requires that a first-order wave history be specified by the user.

A first-order wave history has been generated using the simulate and identify options of WAVEMAKER in Example 1 of Chapters 2 and 3. A first-order wave history which was produced in Chapter 3 and stored in the file **gauss.ide** is used in Chapter 4 to predict waves at other fixed locations. Here, we extend the example to predict how the input wave would be observed from a moving vessel. In this example, the vessel motion time history is an artificially generated sinusoid plus a constant, and is defined in the file **surge.dat**.

The input file for the translation run is:

```
# Wave Prediction Input File

predict gauss.ide 4096 0.5
depth 70.0
translate surge.dat 25 linear
write history gauss.trn ngauss.trn
location 3
0.0
2.0
60.0
```

The input file **gauss.ide** contains a column of 4096 real numbers having an incremental time step of 0.5 seconds (see sample file listed in Appendix E). This file is read in as the first-order components of the identified observed wave history.

The input file **surge.dat** (also listed in Appendix E) contains a column of 4096 real numbers also having an incremental time step of 0.5 seconds. This file is read in as the time history of the vessel motion with positive displacement in the along-wave direction. If the vessel is headed directly into the seas, this file contains the *negative* of the measured surge motion.

Time histories of the wave as observed from the moving vessel are developed for three different observation points on the vessel. The linear (Gaussian) predictions are written to the file **gauss.trn**, while the prediction including the non-linear (non-Gaussian) terms is written to the file **ngauss.trn**.

Results are shown in Figure 5.1 and Figure 5.2. Samples of the wave and motion histories are included in Appendices E and F.

Figure 5.1 shows the total second-order wave history at the original location (the linear part of which was used in the analysis), and the total second order wave as observed from the same point on the moving vessel. The vessel motions are also shown. Notice the wave reaches the fixed observation point prior to reaching the moving observation point when the vessel is displaced in the positive direction (downstream), and reaches the moving observation point prior to the fixed observation point when the vessel is displaced in the negative (upstream) direction.

Figure 5.2 shows the results in ngauss.trn. All three wave histories shown are the total second-order waves as observed from the moving vessel. Notice that the wave shown with no offset is nearly the same shape as the wave shown with a 2 meter offset, and that the main difference between these two wave histories is the phase. The wave offset 60 meters from the original location, however, has a noticeably different shape
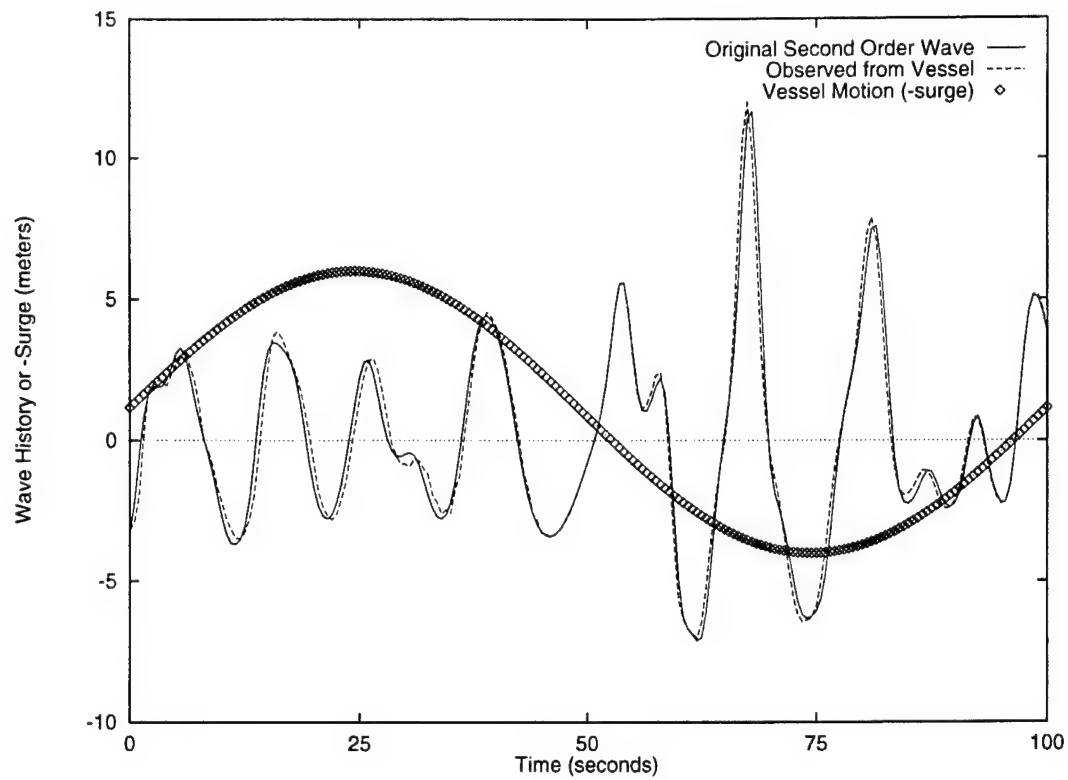
Figure 5.1: Second-Order Total Wave observed With and Without Vessel Motions as Observed from Vessel Center

than the wave as originally input, demonstrating the shape changing of traveling ocean waves.
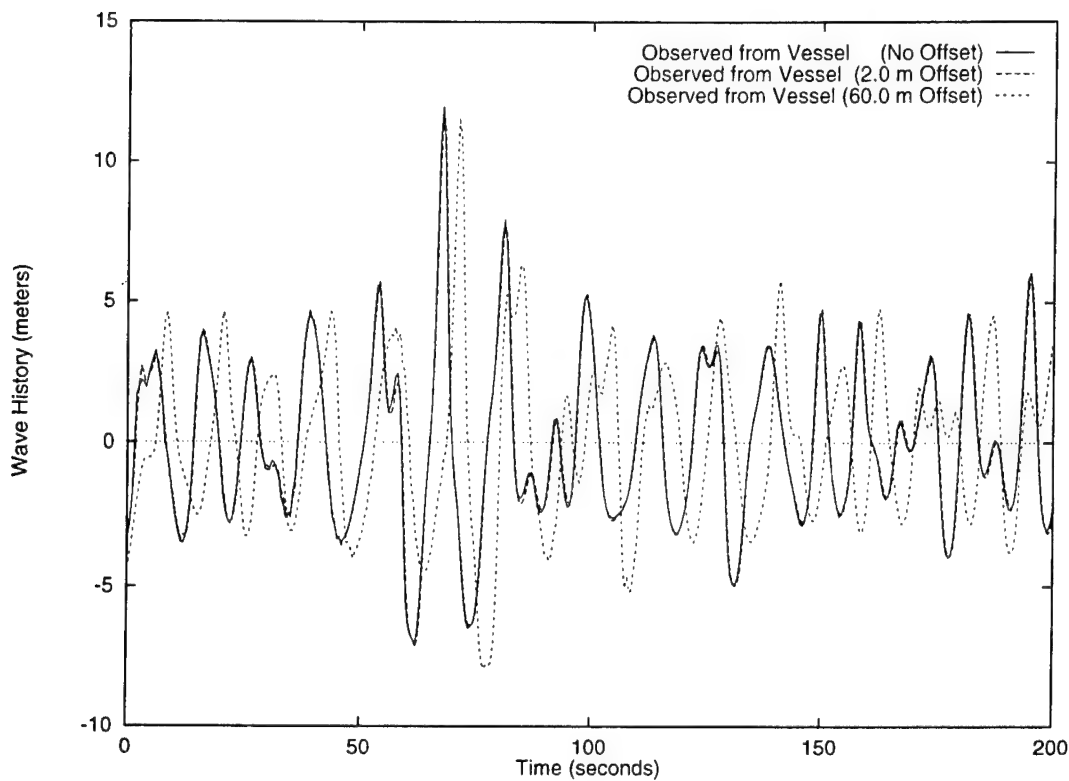
Figure 5.2: Translation Results for three Observation Points on the Moving Vessel
- Second Order Wave

5.3.2   Example 2

The example offered in this section is a natural demonstration of the program using artificially generated wave and vessel motion histories.

A sinusoid with a 10 meter amplitude and a 10 second period has been generated which is used as a first-order wave history. A "sawtooth" pattern has also been generated with a 15 meter amplitude and a 50 second period. These artificially generated wave and motion histories are shown in Figure 5.3.

The original sinusoid has only one frequency; that frequency corresponds to the 10 second wave period. When the original sinusoid is observed from a vessel whose motion corresponds to the "sawtooth" pattern, the observed wave initially is a sinusoid whose frequency is decreased by the vessel's motion in the direction of to the waves, and then becomes another sinusoid whose frequency is increased from that of the original wave by the vessel's motion into that of the waves. Figure 5.4 shows the original wave history, the history as observed from the vessel and the vessel motion
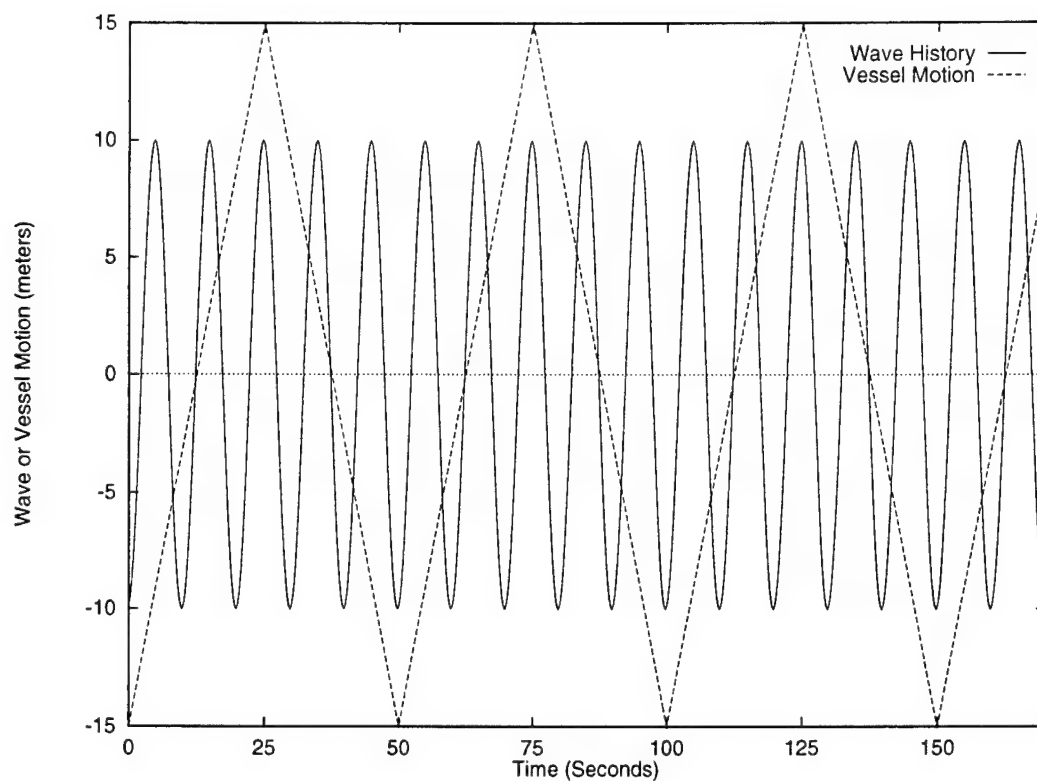
Figure 5.3: Sinusoidal Wave History with Sawtooth Vessel Motion History

history for one complete vessel motion cycle. Notice that the original and observed waves coincide when the vessel is crossing zero offset, and that the difference between the two wave traces is maximum when the vessel is farthest from zero offset.
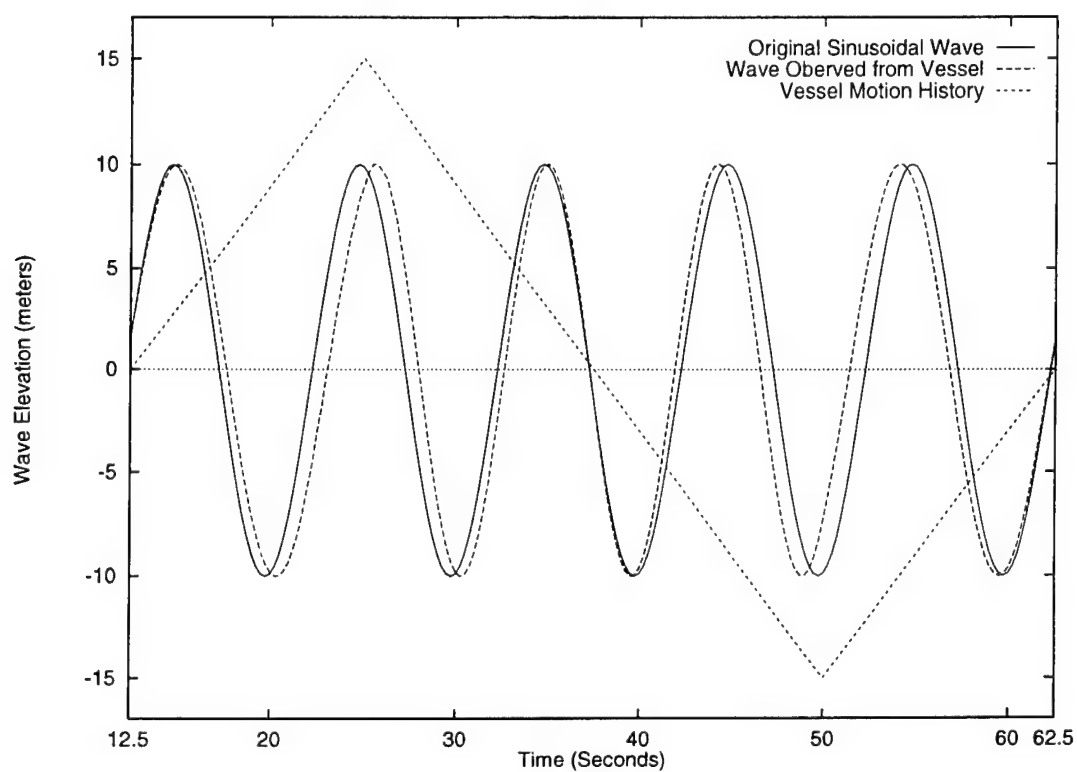
Figure 5.4: Effect of Vessel Movement on Observed Wave History

Chapter 6

Distribution

The WAVEMAKER routine and example files have been distributed on a DOS formatted 3.5 inch floppy diskette. The diskette contains the source code files (of the form *.f), example input (*.inp, and hist.dat), and output files (*.sim, *.sst, *.ide, *.pre, *.pst, *.trn, and *.tst).

6.1 Copying the Diskette

Copy the contents of the diskette on to your host computer (computer on which you will run WAVEMAKER). After the copying is done, your host computer should have:

- Example input files identify.inp, predict.inp, simulate.inp, translate.inp and hist.dat.
- Example output files gauss.sim, ngauss.sim, gauss.sst, ngauss.sst, gauss.ide, ngauss.ide, gauss.pre, ngauss.pre, gauss.trn, and ngauss.trn, in addition to simulate.out, identify.out, predict.out, and translate.out
- manual.ps containing this manual in postscript format
- all the source files *.f and Makefile

Table 6.1 shows the input and output files that are relevant to the simulation and identification examples; Table 6.2 shows those files relevant to the prediction and translation examples.

Table 6.1: Distributed Files for Simulation and Identification Examples

Simulation Example		
File Type	**File Name**	**Description**
Input File	`simulate.inp`	Input to `WAVEMAKER`
Output Files	`simulate.out`	Output from simulate example which would normally be written to the screen
	`gauss.sim`	Simulated first-order wave histories at each specified location
	`ngauss.sim`	Simulated combined first- and second-order wave histories at each specified location
	`gauss.sst`	Statistics of the simulated first-order histories at each specified location
	`ngauss.sst`	Statistics of the simulated combined histories at each specified location
Identification Example		
File Type	**File Name**	**Description**
Input Files	`identify.inp`	Input to `WAVEMAKER`
	`hist.dat`	Observed wave history for which underlying first-order history is to be identified
Output Files	`identify.out`	Output from identify example which would normally be written to the screen
	`gauss.ide`	Identified first-order wave history
	`ngauss.ide`	Identified second-order, combined first- and second-order, and observed wave histories

6.2   Compiling the Source

On a Unix workstation, the `Makefile` can be used to build the `WAVEMAKER` executable. To compile on your host Computer:

- change directory to the subdirectory containing the source code files
- type "make wavmkr" (without the double quotes) at the Unix prompt and press return

The above will compile all the files listed in `Makefile` and link them to make the executable `wavmkr`. Note that the executable `wavmkr` is still in the current directory, and may be moved to the directory which contains the input file, or you can specify

the path to the executable file in order to run it from other directories.

On other operating systems and architectures, follow whatever is the standard procedure for compiling and linking FORTRAN source code distributed over multiple files.

WAVEMAKER has been developed on a Sun Sparcstation2, using version 4.1 of the Sun FORTRAN compiler. Every attempt has been made to adhere to the ANSI FORTRAN 77 standard, ensuring portability of the code.

6.3   Executing the Routine

At the Unix prompt type "wavmkr < simulate.inp" (without the double quotes) in order to execute WAVEMAKER and perform the simulation example analysis. The program reads input from the standard logical input unit. The logical units for standard input, standard output, and standard error are all used in WAVEMAKER. On the Sun compiler, 0 is used for standard error, 5 is standard input and 6 is standard output. If the appropriate unit numbers are different, they can be set using the IOER, IOIN, and IOOU variables in the WAVEMAKER driver program, and the package can be recompiled.

Run the example problem using the complied code to check if you get the same output as provided in the example output files. Note that wavmkr will overwrite any existing file if you specify its name as the target output file using the write command in the input file, or if you do not specify any output file names in the input file.

Similarly, at the Unix prompt type "wavmkr < identify.inp" (without the double quotes) in order to execute WAVEMAKER and perform the identification example analysis. The resulting output histories can be compared to the corresponding example output histories to verify successful compilation of WAVEMAKER.

At the Unix prompt, type "wavmkr < predict.inp" (without the double quotes) in order to execute WAVEMAKER and perform the prediction example analysis. The resulting output histories can again be compared to the corresponding example output histories to verify successful compilation of WAVEMAKER.

The final phase of the example is the translate option, which is executed by "wavmkr < translate.inp." Again, the resulting output histories can be compared to the corresponding example output histories to verify successful compilation of WAVEMAKER.

Table 6.2: Distributed Files for Prediction and Translation Examples

Prediction Example		
File Type	**File Name**	**Description**
Input Files	`predict.inp`	Input to `WAVEMAKER`
	`hist.dat`	Observed wave history for which underlying first-order history is to be identified
Output Files	`predict.out`	Output from predict example which would normally be written to the screen
	`gauss.pre`	Predicted first-order wave history
	`ngauss.pre`	Predicted combined first- and second-order wave histories at each specified location
	`gauss.pst`	Statistics of the simulated first-order histories at each specified location
	`ngauss.pst`	Statistics of the simulated second-order histories at the specified location
Translation Example		
File Type	**File Name**	**Description**
Input Files	`translate.inp`	Input to `WAVEMAKER`
	`surge.dat`	Observed wave history for which underlying first-order history is to be identified
	`gauss.ide`	Identified first-order wave history
Output Files	`translate.out`	Output from translate example which would normally be written to the screen
	`gauss.trn`	Translated first-order wave history
	`ngauss.trn`	Translated combined first- and second-order wave histories at each specified location
	`gauss.tst`	Statistics of the translated first-order histories at each specified location
	`ngauss.tst`	Statistics of the translated second-order histories at the specified location

List of References

Bitner-Gregerson, E., and Haver, S. 1991. Joint environmental model for reliability calculations. *Pages 472–478 of: Proceedings of the 1st international offshore and polar engineering conference (ISOPE)*.

Borgman, L. E. 1969. Ocean wave simulation for engineering design. *Journal of waterways, harbors division*, **95**(4), 556–583. ASCE.

Jha, A. K., and Winterstein, S. R. 1995. *Second-order random waves: Simulation in time and space*. Tech. rept. RMS-17. Reliability of Marine Structures Program, Stanford University, Dept. of Civil Engineering.

Langley, R. S. 1987. A statistical analysis of non-linear random waves. *Ocean engineering (pergamon)*, **14**(5), 389–407.

Longuet-Higgins, M. S. 1963. The effect of non-linearities on statistical distributions in the theory of sea waves. *Journal of fluid mechanics*, **17**(3), 459–480.

Marthinsen, T., and Winterstein, S. R. 1992. On the skewness of random surface waves. *Pages 472–478 of: Proceedings of the 2nd international offshore and polar engineering conference, san francisco*. ISOPE.

Molin, B, and Chen, X. B. 1990. *Calculation of second-order sum-frequency loads on tlp hulls*. Tech. rept. Institute Français du Petrele Report.

SWIM, 2.1. 1995. *SWIM: Slow wave-induced motions– user's manual*. Dept. of Ocean Engineering, M.I.T.

Ude, T. C. 1994. *Second-order load and response models for floating structures: Probabilistics analysis and system identification*. Tech. rept. RMS–16. Reliability of Marine Structures, Dept. of Civil Engr., Stanford University.

Ude, T. C., and Winterstein, S. R. 1996. Calibration of models for slow drift motions using statistical moments of observed data. *In: 6th international offshore and polar engineering conference ISOPE*. to appear.

Vinje, T., and Haver, S. 1994. On the non-Gaussian structure of ocean waves. *Pages 453–480 of: Proceedings of the 7th international conference on the behaviour of offshore structures (BOSS)*, vol. 2.

WAMIT, 4.0. 1995. *WAMIT: A radiation-diffraction panel program for wave-body interactions–user's manual.* Dept. of Ocean Engineering, M.I.T.

Winterstein, S. R., and Jha, A. K. 1995. Random models of second-order waves and local wave statistics. *Pages 1171–1174 of: Proceedings of the 10th engineering mechanics speciality conference.* ASCE.

Jha, A. K., and Winterstein, S. R. 1995. *Second-order random waves: Simulation in time and space.* Tech. rept. RMS-17. Reliability of Marine Structures Program, Stanford University, Dept. of Civil Engineering.

Jha, A. K., and Winterstein, S. R. 1996. *Second-order random waves: Simulation and Identification of Second Order Random Waves.* Tech. rept. RMS-22. Reliability of Marine Structures Program, Stanford University, Dept. of Civil Engineering.

Sweetman, J. A., Jha, A. K., and Winterstein, S. R. 1998. *Second-order random waves: Prediction of Temporal and Spatial Variation.* Tech. rept. RMS-33. Reliability of Marine Structures Program, Stanford University, Dept. of Civil Engineering.

Sweetman, J. A., and Winterstein, S. R. 1998. *Second-Order Random Waves: Translation to a Moving Reference Frame.* Tech. note TN-5. Reliability of Marine Structures Program, Stanford University, Dept. of Civil Engineering.

Appendix A

Output Files for Simulation Example

Output File: **gauss.sim**

```
# Underlying First-Order Wave Process
#                 Wave Elevation at Spatial Location
#   Time(sec.)      0.00     60.00
       0.000000    -2.888    -4.724
       0.500000    -1.533    -5.202
       1.000000    -0.235    -5.124
       1.500000     0.735    -4.605
       2.000000     1.281    -3.795
       2.500000     1.495    -2.834
       3.000000     1.581    -1.830
       3.500000     1.739    -0.865
       4.000000     2.062    -0.013
       4.500000     2.495     0.661
       5.000000     2.877     1.106
       5.500000     3.027     1.322
       6.000000     2.839     1.377
       6.500000     2.322     1.402
       7.000000     1.592     1.548
       7.500000     0.798     1.910
       8.000000     0.049     2.470
       8.500000    -0.628     3.069
       9.000000    -1.284     3.453
       9.500000    -1.983     3.370
      10.000000    -2.736     2.684
      10.500000    -3.456     1.455
      11.000000    -3.971    -0.061
      11.500000    -4.088    -1.496
      12.000000    -3.678    -2.512
      12.500000    -2.744    -2.928
      13.000000    -1.434    -2.791
      13.500000     0.005    -2.346
          :           :         :
          :           :         :
          :           :         :
    2046.000000    -3.491     1.530
    2046.500000    -4.228    -0.291
    2047.000000    -4.427    -2.115
    2047.500000    -3.948    -3.664
```

Note that ":" indicates more numbers. Since the files are long, we present truncated versions of the output files.

Output File: **ngauss.sim**

```
# Total Second-Order Process
#                 Wave Elevation at Spatial Location
#  Time(sec.)     0.00    60.00
     0.000000    -3.318   -4.515
     0.500000    -2.181   -4.669
     1.000000    -0.625   -4.621
     1.500000     0.788   -4.309
     2.000000     1.624   -3.706
     2.500000     1.890   -2.876
     3.000000     1.890   -1.933
     3.500000     1.918   -0.972
     4.000000     2.144   -0.061
     4.500000     2.587    0.727
     5.000000     3.068    1.301
     5.500000     3.279    1.604
     6.000000     3.005    1.680
     6.500000     2.316    1.671
     7.000000     1.484    1.738
     7.500000     0.731    2.010
     8.000000     0.099    2.552
     8.500000    -0.500    3.284
     9.000000    -1.170    3.852
     9.500000    -1.942    3.719
    10.000000    -2.721    2.593
    10.500000    -3.329    0.799
    11.000000    -3.643   -0.907
    11.500000    -3.682   -1.969
    12.000000    -3.502   -2.398
    12.500000    -3.024   -2.494
    13.000000    -2.072   -2.399
    13.500000    -0.627   -2.098
        :           :        :
        :           :        :
        :           :        :
  2046.000000    -3.626    0.435
  2046.500000    -3.966   -1.679
  2047.000000    -3.995   -3.226
  2047.500000    -3.845   -4.105
```

Output File: **gauss.sst**

```
# Underlying First-Order Wave Process
# Location      Mean      Sigma      Skewness    Kurtosis    Minimum    Maximum
        0.00  -.1972E-10  2.967      0.3419E-01  2.908       -9.017      10.26
       60.00  -.1251E-08  2.967      0.1990E-01  2.916       -9.264      9.900
```

Output File: **ngauss.sst**

```
# Total Second-Order Process
# Location      Mean      Sigma      Skewness    Kurtosis    Minimum    Maximum
        0.00  0.4132E-10  2.993      0.2103      2.971       -8.142      11.67
       60.00  -.1501E-08  3.000      0.1948      2.974       -8.335      11.45
```

Appendix B

Input File for Identification Example

Input File: **hist.dat**

```
-3.318
-2.181
-0.625
 0.788
 1.624
 1.890
 1.890
 1.918
 2.144
 2.587
 3.068
 3.279
 3.005
 2.316
 1.484
 0.731
 0.099
-0.500
-1.170
-1.942
-2.721
-3.329
-3.643
-3.682
-3.502
-3.024
-2.072
-0.627
 1.009
 2.380
 3.196
   :
   :
   :
-2.816
-3.626
-3.966
-3.995
-3.845
```

Note that ":" indicates more numbers. Since the files are long, we present truncated versions of the output files.

Appendix C

Output Files for Identification Example

Output File: **gauss.ide**

```
# Identified First-Order Wave History
   -2.9255
   -1.5714
   -0.2291
    0.6794
    1.2878
    1.4340
    1.5983
    1.7467
    2.0913
    2.5642
    2.8681
    2.9942
    2.8671
    2.3110
    1.5875
    0.7704
    0.0445
   -0.6462
   -1.2719
   -1.9948
   -2.7282
   -3.4661
   -3.9602
   -4.0758
   -3.6428
   -2.7268
   -1.4023
    0.0159
    1.3349
    2.3032
      :
      :
      :
   -2.2371
   -3.0895
   -3.8786
   -4.1455
   -4.2195
```

Output File: **ngauss.ide**

```
# Identified 2nd-Order Wave, Identified Total Wave, Input Wave History
   -0.3925    -3.3180    -3.3180
   -0.6096    -2.1810    -2.1810
   -0.3959    -0.6250    -0.6250
    0.1086     0.7880     0.7880
    0.3362     1.6240     1.6240
    0.4560     1.8900     1.8900
    0.2917     1.8900     1.8900
    0.1713     1.9180     1.9180
    0.0527     2.1440     2.1440
    0.0228     2.5870     2.5870
    0.1999     3.0680     3.0680
    0.2848     3.2790     3.2790
    0.1379     3.0050     3.0050
    0.0050     2.3160     2.3160
   -0.1035     1.4840     1.4840
   -0.0394     0.7310     0.7310
    0.0545     0.0990     0.0990
    0.1462    -0.5000    -0.5000
    0.1019    -1.1700    -1.1700
    0.0528    -1.9420    -1.9420
    0.0072    -2.7210    -2.7210
    0.1371    -3.3290    -3.3290
    0.3172    -3.6430    -3.6430
    0.3938    -3.6820    -3.6820
    0.1408    -3.5020    -3.5020
   -0.2972    -3.0240    -3.0240
   -0.6697    -2.0720    -2.0720
   -0.6429    -0.6270    -0.6270
   -0.3259     1.0090     1.0090
    0.0768     2.3800     2.3800
    0.2842     3.1960     3.1960
      :          :          :
      :          :          :
      :          :          :
   -0.5789    -2.8160    -2.8160
   -0.5365    -3.6260    -3.6260
   -0.0874    -3.9660    -3.9660
    0.1505    -3.9950    -3.9950
    0.3745    -3.8450    -3.8450
```

Appendix D

Output Files for Prediction Example

Output File: **gauss.pre**

```
# Underlying First-Order Wave Process
#                 Wave Elevation at Spatial Location
#   Time(sec.)      0.00      2.00     60.00
     0.000000     -2.925    -3.104    -4.630
     0.500000     -1.572    -2.127    -3.977
     1.000000     -0.229    -0.747    -3.263
     1.500000      0.679     0.506    -2.533
     2.000000      1.288     1.355    -1.882
     2.500000      1.434     1.582    -1.389
     3.000000      1.599     1.778    -1.128
     3.500000      1.746     1.745    -1.088
     4.000000      2.092     1.999    -1.162
     4.500000      2.564     2.311    -1.143
     5.000000      2.869     2.699    -0.843
     5.500000      2.994     2.958    -0.146
     6.000000      2.868     2.858     0.889
     6.500000      2.311     2.499     2.097
     7.000000      1.588     1.881     3.224
     7.500000      0.770     0.968     3.999
     8.000000      0.045     0.172     4.208
     8.500000     -0.647    -0.581     3.807
     9.000000     -1.271    -1.218     2.925
     9.500000     -1.995    -1.860     1.822
    10.000000     -2.728    -2.508     0.778
    10.500000     -3.467    -3.194    -0.020
    11.000000     -3.960    -3.737    -0.548
    11.500000     -4.076    -3.998    -0.892
    12.000000     -3.642    -3.785    -1.184
    12.500000     -2.727    -3.049    -1.525
    13.000000     -1.402    -1.867    -1.960
    13.500000      0.015    -0.444    -2.390
        :           :         :         :
        :           :         :         :
        :           :         :         :
  2045.500000     -2.238    -1.883     3.165
  2046.000000     -3.089    -2.727     1.545
  2046.500000     -3.879    -3.668    -0.338
  2047.000000     -4.145    -4.054    -2.207
  2047.500000     -4.220    -4.517    -3.778
```

Output File: **ngauss.pre**

```
# Total Second-Order Process
#             Wave Elevation at Spatial Location
#  Time(sec.)      0.00       2.00      60.00
      0.000000    -3.531     -3.125     -4.348
      0.500000    -2.938     -2.775     -3.810
      1.000000    -0.895     -2.045     -3.187
      1.500000     0.648     -0.291     -2.582
      2.000000     1.493      1.720     -1.901
      2.500000     1.886      1.937     -1.328
      3.000000     1.905      2.196     -0.920
      3.500000     1.815      1.959     -0.714
      4.000000     2.035      2.048     -0.729
      4.500000     2.530      2.308     -0.792
      5.000000     2.953      2.689     -0.798
      5.500000     3.253      3.203     -0.543
      6.000000     3.129      2.987      0.146
      6.500000     2.302      2.847      1.332
      7.000000     1.637      1.802      2.793
      7.500000     0.809      0.834      4.118
      8.000000     0.115      0.292      4.612
      8.500000    -0.441     -0.437      4.112
      9.000000    -1.115     -1.068      2.930
      9.500000    -1.916     -1.729      1.683
     10.000000    -2.629     -2.298      0.751
     10.500000    -3.279     -3.150      0.147
     11.000000    -3.582     -3.335     -0.298
     11.500000    -3.715     -3.621     -0.648
     12.000000    -3.497     -3.534     -0.967
     12.500000    -3.119     -3.203     -1.383
     13.000000    -2.129     -2.486     -1.861
     13.500000    -0.719     -1.252     -2.258
          :          :          :          :
          :          :          :          :
          :          :          :          :
   2045.500000    -2.856     -2.331      2.296
   2046.000000    -3.539     -3.433     -0.114
   2046.500000    -3.940     -3.722     -2.299
   2047.000000    -3.668     -4.017     -3.792
   2047.500000    -4.098     -4.129     -4.519
```

Output File: **gauss.pst**

```
# Underlying First-Order Wave Process
# Location     Mean    Sigma     Skewness   Kurtosis   Minimum    Maximum
      0.00  0.2512E-01  2.966    0.2018E-01  2.913     -9.087     10.27
      2.00  0.2507E-01  2.966    0.2001E-01  2.913     -9.119     10.29
     60.00  0.2615E-01  2.965    0.8905E-02  2.924     -9.293      9.888
```

Output File: **ngauss.pst**

```
# Total Second-Order Process
# Location     Mean    Sigma     Skewness   Kurtosis   Minimum    Maximum
      0.00  0.2480E-01  2.993     0.1960     2.969     -8.212     11.68
      2.00  0.2481E-01  2.994     0.1956     2.969     -8.249     11.86
     60.00  0.2578E-01  2.999     0.1830     2.979     -8.434     11.72
```

Appendix E

Time History Input Files for Translate Example

Input File: **gauss.ide**

```
# Identified First-Order Wave History
   -2.9255
   -1.5714
   -0.2291
    0.6794
    1.2878
    1.4340
    1.5983
    1.7467
    2.0913
    2.5642
    2.8681
    2.9942
    2.8671
    2.3110
    1.5875
    0.7704
    0.0445
   -0.6462
   -1.2719
   -1.9948
   -2.7282
   -3.4661
   -3.9602
   -4.0758
   -3.6428
   -2.7268
   -1.4023
    0.0159
    1.3349
    2.3032
    2.9118
      :
      :
      :
   -2.2371
   -3.0895
   -3.8786
   -4.1455
   -4.2195
```

Note that this file is the same as the output file from the identify example.

Input File: **surge.dat**

```
       1.1570538e+00
       1.3139526e+00
       1.4705416e+00
       1.6266662e+00
       1.7821723e+00
       1.9369066e+00
       2.0907162e+00
       2.2434494e+00
       2.3949555e+00
       2.5450850e+00
       2.6936896e+00
       2.8406228e+00
       2.9857395e+00
       3.1288965e+00
       3.2699525e+00
       3.4087684e+00
       3.5452071e+00
       3.6791340e+00
       3.8104169e+00
       3.9389263e+00
       4.0645353e+00
       4.1871199e+00
       4.3065593e+00
       4.4227355e+00
       4.5355339e+00
       4.6448431e+00
       4.7505553e+00
       4.8525662e+00
       4.9507751e+00
       5.0450850e+00
       5.1354029e+00
            :
            :
            :
       2.2434494e+00
       2.0907162e+00
       1.9369066e+00
       1.7821723e+00
       1.6266662e+00
```

Appendix F

Output Files for Translate Example

Output File: **gauss.trn**

```
# Underlying First-Order Wave Process
# Waves as observed From Vessel
#               Wave Elevation at Spatial Location
#   Time(sec.)     0.00      2.00     60.00
       0.000000   -3.076    -3.222    -4.747
       0.500000   -1.945    -2.329    -4.102
       1.000000   -0.626    -1.107    -3.400
       1.500000    0.597     0.115    -2.687
       2.000000    1.330     1.234    -2.024
       2.500000    1.586     1.834    -1.474
       3.000000    1.768     1.937    -1.102
       3.500000    1.792     1.916    -0.933
       4.000000    1.934     1.943    -0.914
       4.500000    2.308     2.155    -0.931
       5.000000    2.598     2.384    -0.811
       5.500000    2.883     2.698    -0.397
       6.000000    2.916     2.880     0.364
       6.500000    2.545     2.629     1.407
       7.000000    1.987     2.209     2.536
       7.500000    1.220     1.457     3.511
       8.000000    0.315     0.537     4.103
       8.500000   -0.524    -0.268     4.159
       9.000000   -1.165    -1.124     3.654
       9.500000   -1.769    -1.775     2.690
      10.000000   -2.380    -2.250     1.509
      10.500000   -2.926    -2.772     0.380
      11.000000   -3.477    -3.273    -0.501
      11.500000   -3.809    -3.622    -1.057
      12.000000   -3.811    -3.723    -1.359
      12.500000   -3.361    -3.487    -1.540
      13.000000   -2.408    -2.718    -1.744
      13.500000   -1.134    -1.575    -2.024
          :          :         :         :
          :          :         :         :
          :          :         :         :
    2045.500000   -1.951    -1.809     3.520
    2046.000000   -2.678    -2.372     2.062
    2046.500000   -3.638    -3.217     0.219
    2047.000000   -4.142    -4.026    -1.729
    2047.500000   -4.390    -4.436    -3.471
```

Output File: **ngauss.trn**

```
# Total Second-Order Process
# Waves as observed From Vessel
#              Wave Elevation at Spatial Location
#  Time(sec.)    0.00      2.00     60.00
       0.000000  -3.169    -3.518   -4.430
       0.500000  -2.886    -2.724   -3.927
       1.000000  -2.002    -1.971   -3.316
       1.500000   0.079    -0.979   -2.682
       2.000000   1.689     0.819   -2.057
       2.500000   1.951     2.054   -1.473
       3.000000   2.153     2.630   -0.989
       3.500000   2.093     2.387   -0.655
       4.000000   1.930     1.917   -0.541
       4.500000   2.330     2.302   -0.546
       5.000000   2.656     2.496   -0.535
       5.500000   2.984     2.684   -0.449
       6.000000   3.200     3.141   -0.134
       6.500000   2.663     2.895    0.674
       7.000000   2.131     2.279    1.930
       7.500000   1.242     1.411    3.277
       8.000000   0.020     0.520    4.338
       8.500000  -0.365    -0.288    4.589
       9.000000  -0.897    -1.169    3.873
       9.500000  -1.657    -1.487    2.587
      10.000000  -2.211    -1.997    1.262
      10.500000  -2.790    -2.624    0.269
      11.000000  -3.155    -3.027   -0.386
      11.500000  -3.454    -3.377   -0.770
      12.000000  -3.478    -3.189   -1.059
      12.500000  -3.226    -3.294   -1.310
      13.000000  -2.751    -2.950   -1.551
      13.500000  -1.944    -2.074   -1.883
          :         :         :        :
          :         :         :        :
          :         :         :        :
    2045.500000  -2.108    -1.718    3.058
    2046.000000  -3.421    -2.797    0.684
    2046.500000  -3.722    -3.466   -1.675
    2047.000000  -4.088    -4.333   -3.527
    2047.500000  -3.983    -3.967   -4.516
```

Output File: **gauss.tst**

```
# Underlying First-Order Wave Process
# Location     Mean     Sigma     Skewness    Kurtosis    Minimum    Maximum
# Waves as observed From Vessel
      0.00   0.2521E-01   2.964    0.1816E-01   2.922     -9.141     10.41
      2.00   0.2520E-01   2.964    0.1842E-01   2.925     -9.153     10.34
     60.00   0.2618E-01   2.962    0.1356E-01   2.931     -9.390      9.957
```

Output File: **ngauss.tst**

```
# Total Second-Order Process
# Location     Mean     Sigma     Skewness    Kurtosis    Minimum    Maximum
# Waves as observed From Vessel
      0.00   0.2514E-01   2.990     0.1902      2.979     -8.245     11.97
      2.00   0.2509E-01   2.990     0.1897      2.983     -8.260     11.83
     60.00   0.2615E-01   2.996     0.1859      2.992     -8.429     11.46
```

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*
00-09-1999

2. REPORT DATE
May - 1999

3. DATES COVERED *(From - To)*

4. TITLE AND SUBTITLE
Second-Order Random Ocean Waves: Prediction of Temporal and Spatial Variation From Fixed and Moving References.

5a. CONTRACT NUMBER

5b. GRANT NUMBER
N00014-96-1-0641

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Bert Sweetman
Steven R. Winterstein

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
RMS Group
S. R. Winterstein and C. A. Cornell
Blume Center
Stanford University, CA 94305

8. PERFORMING ORGANIZATION REPORT NUMBER
RMS-37

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Office of Naval Research
800 N. Quincy St.
Arlington, VA 22217-4620
ATTN: Dr. Roshdy Barsoum

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSORING/MONITORING AGENCY REPORT NUMBER

12. DISTRIBUTION AVAILABILITY STATEMENT
Approved For Public Release

13. SUPPLEMENTARY NOTES

14. ABSTRACT This report describes and illustrates the use and underlying theory of the routine Wavemaker. The Fortran computer program is used to simulate random Non-Gaussian ocean wave time-histories, to identify first- and second-order components of user specified waves, and to predict wave time-histories at user-specified fixed or moving spatial locations based on the originally specified or simulated wave time-history. Ocean wave histories are simulated using a first-order wave process and arbitrary power spectrum and applying Non-linear corrections based on second-order hydrodynamics (supercedes RMS 33)

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:

a. REPORT	b. ABSTRACT	c. THIS PAGE

17. LIMITATION OF ABSTRACT

18. NUMBER OF PAGES

19a. NAME OF RESPONSIBLE PERSON

19b. TELEPONE NUMBER *(Include area code)*

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI-Std Z39-18